

TradeApi.History Namespace

TE API

► Classes

Class	Description
 BarData	BarData entity
 BBOData	Historical best bid offer entity
 ClusterHistoricalRequest	Cluster data based historical request
 CustomData	Custom data entity
 HeikinAshiHistoricalRequest	HeikinAshi based historical request
 HistoricalData	Historical data entity
 HistoricalDataManager	Historical data manager entity
 HistoricalRequest	Historical Request
 Interval	Represents Datetime limits for

		historical data queries
❖	KagiHistoricalRequest	Kagi based historical request
❖	LineBreakHistoricalRequest	LineBreak data based historical request
❖	PointAndFigureHistoricalRequest	PointAndFigure data based historical request
❖	PriceRangeHistoricalRequest	PriceRange based historical request
❖	ProfileHistoricalRequest	Profile data based historical request
❖	RenkoHistoricalRequest	Renko data based historical request
❖	TickHistoricalRequest	Tick based historical request
❖	TimeHistoricalRequest	Time based historical request
❖	TradeData	Trade data entity
❖	VolumeHistoricalRequest	Volume data

based
historical
request

▪ Enumerations

Enumeration	Description
 BuildFrom	Type of the source data from which is build
 ClusterDataType	Type of the ClusterData
 DataType	Supported history data types
 Period	Period of data
 PointAndFigureStyle	Type of the PnF
 PriceType	Type of the price
 RenkoStyle	Type of the Renko

BarData Class

BarData entity

► Inheritance Hierarchy

SystemObject TradeApi.HistoryHistoricalData
TradeApi.HistoryBarData

Namespace: TradeApi.History

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class BarData :  
    HistoricalData
```

[Copy](#)

The `BarData` type exposes the following members.

► Properties

	Name	Description
	Count	The count of the historical data elements (Inherited from HistoricalData)
	HistoricalRequest	Reference to the initial HistoricalRequest of current HistoricalData (Inherited from HistoricalData)

[Top](#)

Methods

Name	Description
Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindInterval	Index of the historical unit by given time in UTC (Inherited from HistoricalData)
GetClose	Gets low price of the bar by offset
GetHashCode	Serves as the default hash function. (Inherited from Object)
GetHigh	Gets high price of the bar by offset
GetHighest	A collection of the unit indexes which are the highest values within interval (Inherited from HistoricalData)
GetLow	Gets low price of the bar by offset
GetLowest	A collection of the unit indexes which are the lowest values within interval (Inherited from HistoricalData)
GetOpen	Gets an open price of the bar by offset
GetTicks	Amount of ticks per given historical bar
GetTimeUtc	Amount of ticks per given historical bar

(Overrides
[HistoricalDataGetTimeUtc\(Int32\)](#))

≡ [GetType](#) Gets the [Type](#) of the current instance.
(Inherited from [Object](#))

≡ [GetValue](#) Retrieves historical value by index based on price type
(Inherited from [HistoricalData](#))

≡ [GetVolume](#) Volume of Ticks for Bid/Ask based Historical data, or volume of Trades for Trade based Historical data

≡ [ToString](#) Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

See Also

[Reference](#)

[TradeApi.History Namespace](#)

BarData Properties

The [BarData](#) type exposes the following members.

► Properties

Name	Description
 Count	The count of the historical data elements (Inherited from HistoricalData)
 HistoricalRequest	Reference to the initial HistoricalRequest of current HistoricalData (Inherited from HistoricalData)

[Top](#)

► See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BarData Methods

The [BarData](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	FindInterval	Index of the historical unit by given time in UTC (Inherited from HistoricalData)
	GetClose	Gets low price of the bar by offset
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetHigh	Gets high price of the bar by offset
	GetHighest	A collection of the unit indexes which are the highest values within interval (Inherited from HistoricalData)
	GetLow	Gets low price of the bar by offset
	GetLowest	A collection of the unit indexes which are the lowest values within interval (Inherited from HistoricalData)
	GetOpen	Gets an open price of the bar by

	offset	
≡	GetTicks	Amount of ticks per given historical bar
≡	GetTimeUtc	Amount of ticks per given historical bar (Overrides HistoricalDataGetTimeUtc(Int32))
≡	GetType	Gets the Type of the current instance. (Inherited from Object)
≡	GetValue	Retrieves historical value by index based on price type (Inherited from HistoricalData)
≡	GetVolume	Volume of Ticks for Bid/Ask based Historical data, or volume of Trades for Trade based Historical data
≡	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BarDataGetClose Method

Gets low price of the bar by offset

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetClose(  
    int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bar's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{
```

[Copy](#)

```
        public class
GetCloseExample :
StrategyBuilder
{
    public
GetCloseExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetCloseExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
HistoryDataSeries.Count > 1)
    {
        var barData
= HistoryDataSeries as
BarData;
        var close =
barData.GetClose(1);

Notification.Print($"Last
bar's close price is
{close}");
    }
}
```

```
        }  
    }
```

See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BarDataGetHigh Method

Gets high price of the bar by offset

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetHigh(  
    int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bar's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
GetHighExample :
StrategyBuilder
{
    public
GetHighExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetHighExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
HistoryDataSeries.Count > 1)
    {
        var barData
= HistoryDataSeries as
BarData;
        var high =
barData.GetHigh(1);

Notification.Print($"Last
bar's high price is {high}");
    }
}
```

```
        }  
    }
```

See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BarDataGetLow Method

Gets low price of the bar by offset

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetLow(  
    int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bar's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
GetLowExample :
StrategyBuilder
{
    public
GetLowExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetLowExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
HistoryDataSeries.Count > 1)
    {
        var barData
= HistoryDataSeries as
BarData;
        var low =
barData.GetLow(1);

Notification.Print($"Last
bar's low price is {low}");
    }
}
```

```
        }  
    }
```

See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BarDataGetOpen Method

Gets an open price of the bar by offset

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetOpen(  
    int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bar's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
GetOpenExample :
StrategyBuilder
{
    public
GetOpenExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetOpenExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
HistoryDataSeries.Count > 1)
    {
        var barData
= HistoryDataSeries as
BarData;
        var open =
barData.GetOpen(1);

Notification.Print($"Last
bar's open price is {open}");
    }
}
```

```
        }  
    }
```

See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BarDataGetTicks Method

Amount of ticks per given historical bar

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetTicks(  
    int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bar's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
GetTicksExample :
StrategyBuilder
{
    public
GetTicksExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"GetTicksExample";
    #endregion
}
    public BarData
barData;

    public override
void Init()
{
    this.barData =
HistoryDataSeries as BarData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
BarData) && this.barData.Count
> 1)
    {
        var ticks =
barData.GetTicks(1);

Notification.Print($"Last
bar's ticks is {ticks}");
    }
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BarDataGetTimeUtc Method

Amount of ticks per given historical bar

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override DateTime
GetTimeUtc(
    int offset = 0
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bar's index offset

Return Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
```

[Copy](#)

```
    {
        public class
GetTimeUtcExample :
StrategyBuilder
{
        public
GetTimeUtcExample()
: base()
{
            #region
Initialization

Credentials.ProjectName =
"GetTimeUtcExample";
            #endregion
}
        public BarData
barData;

        public override
void Init()
{
            this.barData =
HistoryDataSeries as BarData;
}

        public override
void Update(TickStatus args)
{
            if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
BarData) && this.barData.Count
> 1)
{
            var time =
barData.GetTimeUtc(1);

Notification.Print($"Last
bar's time is {time}");
}
```

```
        }  
    }  
}
```

See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BarDataGetVolume Method

Volume of Ticks for Bid/Ask based Historical data, or volume of Trades for Trade based Historical data

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetVolume(Copy
    int offset = 0
)
```

Parameters

offset [Int32](#) (Optional)
bar's index offset

Return Value
[Double](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
using TradeApi.History;
```

```
namespace TestEnv
{
    public class GetVolumeExample : StrategyBuilder
    {
        public GetVolumeExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "GetVolumeExample";
            #endregion
        }
        public BarData barData;
        public override void Init()
        {
            this.barData =
                HistoryDataSeries as BarData;
        }

        public override void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar &&
                (this.HistoryDataSeries is BarData) && this.barData.Count
                > 1)
            {
                var volume
                    = barData.GetVolume(1);
                Notification.Print($"Last
```

```
    bar's volume is {volume}");  
    }  
}  
}
```

See Also

[Reference](#)

[BarData Class](#)

[TradeApi.History Namespace](#)

BBOData Class

Historical best bid offer entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalData](#)
[TradeApi.HistoryBBOData](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class BBOData :  
    HistoricalData
```

[Copy](#)

The [BBOData](#) type exposes the following members.

► Properties

	Name	Description
	Count	The count of the historical data elements (Inherited from HistoricalData)
	HistoricalRequest	Reference to the initial HistoricalRequest of current HistoricalData (Inherited from HistoricalData)

[Top](#)

Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 FindInterval	Index of the historical unit by given time in UTC (Inherited from HistoricalData)
 GetAsk	The ask price of the required Historical tick
 GetAskSize	The ask size of the required Historical tick
 GetBid	The bid price of the required Historical tick
 GetBidSize	The bid size of the required Historical tick
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetHighest	A collection of the unit indexes which are the highest values within interval (Inherited from HistoricalData)
 GetLowest	A collection of the unit indexes which are the lowest values within interval (Inherited from HistoricalData)
 GetTimeUtc	The volume of the required Historical tick (Overrides HistoricalDataGetTimeUtc(Int32))

	GetType	Gets the Type of the current instance. (Inherited from Object)
 	GetValue	Retrieves historical value by index based on price type (Inherited from HistoricalData)
 	GetVolume	The volume of the required Historical tick
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

[TradeApi.History Namespace](#)

BBOData Properties

The [BBOData](#) type exposes the following members.

► Properties

Name	Description
 Count	The count of the historical data elements (Inherited from HistoricalData)
 HistoricalRequest	Reference to the initial HistoricalRequest of current HistoricalData (Inherited from HistoricalData)

[Top](#)

► See Also

[Reference](#)

[BBOData Class](#)

[TradeApi.History Namespace](#)

BBOData Methods

The [BBOData](#) type exposes the following members.

▪ Methods

Name	Description
Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindInterval	Index of the historical unit by given time in UTC (Inherited from HistoricalData)
GetAsk	The ask price of the required Historical tick
GetAskSize	The ask size of the required Historical tick
GetBid	The bid price of the required Historical tick
GetBidSize	The bid size of the required Historical tick
GetHashCode	Serves as the default hash function. (Inherited from Object)
GetHighest	A collection of the unit indexes which are the highest values within interval (Inherited from HistoricalData)
GetLowest	A collection of the unit indexes which are the lowest values

		within interval (Inherited from HistoricalData)
 	GetTimeUtc	The volume of the required Historical tick (Overrides HistoricalData.GetTimeUtc(Int32))
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	GetValue	Retrieves historical value by index based on price type (Inherited from HistoricalData)
 	GetVolume	The volume of the required Historical tick
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

[BBOData Class](#)

[TradeApi.History Namespace](#)

BBODataGetAsk Method

The ask price of the required Historical tick

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetAsk(  
    int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bbo's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
GetAskExample : 
StrategyBuilder
{
    public
GetAskExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetAskExample";
    #endregion
}
    public BBOData
bbodata;

    public override
void Init()
{
    this.bboData =
HistoryDataSeries as BBOData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
BBOData) && this.bboData.Count
> 1)
    {
        var ask =
bbodata.GetAsk(1);

Notification.Print($"Last
tick's ask is {ask}");
    }
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[BBOData Class](#)

[TradeApi.History Namespace](#)

BBODataGetAskSize Method

The ask size of the required Historical tick

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetAskSize(Copy
                          int offset = 0
)
```

Parameters

offset [Int32](#) (Optional)
bbo's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
```

```
    public class
GetAskSizeExample :
StrategyBuilder
{
    public
GetAskSizeExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"GetAskSizeExample";
    #endregion
}
    public BBOData
bbodata;

    public override
void Init()
{
    this.bboData =
HistoryDataSeries as BBOData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
BBOData) && this.bboData.Count
> 1)
    {
        var askSize
= bboData.GetAskSize(1);

Notification.Print($"Last
tick's ask size is
{askSize}");
}
```

```
        }  
    }  
}
```

See Also

[Reference](#)

[BBOData Class](#)

[TradeApi.History Namespace](#)

BBODataGetBid Method

The bid price of the required Historical tick

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetBid(  
    int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bbo's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
GetBidExample : 
StrategyBuilder
{
    public
GetBidExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetBidExample";
    #endregion
}
    public BBOData
bbodata;

    public override
void Init()
{
    this.bboData =
HistoryDataSeries as BBOData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
BBOData) && this.bboData.Count
> 1)
    {
        var bid =
bbodata.GetBid(1);

Notification.Print($"Last
tick's bid is {bid}");
    }
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[BBOData Class](#)

[TradeApi.History Namespace](#)

BBODataGetBidSize Method

The bid size of the required Historical tick

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetBidSize(Copy
                          int offset = 0
                        )
```

Parameters

offset [Int32](#) (Optional)
bbo's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
```

```
    public class
GetBidSizeExample :
StrategyBuilder
{
    public
GetBidSizeExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"GetBidSizeExample";
    #endregion
}
    public BBOData
bboData;

    public override
void Init()
{
    this.bboData =
HistoryDataSeries as BBOData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
BBOData) && this.bboData.Count
> 1)
    {
        var bidSize
= bboData.GetBidSize(1);

Notification.Print($"Last
tick's bid size is
{bidSize}");
}
```

```
        }  
    }  
}
```

See Also

[Reference](#)

[BBOData Class](#)

[TradeApi.History Namespace](#)

BBODataGetTimeUtc Method

The volume of the required Historical tick

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override DateTime
GetTimeUtc(
    int offset = 0
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
bbo's index offset

Return Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
```

[Copy](#)

```
    {
        public class
GetTimeUtcExample : 
StrategyBuilder
{
        public
GetTimeUtcExample()
: base()
{
            #region
Initialization

Credentials.ProjectName =
"GetTimeUtcExample";
            #endregion
}
        public BBOData
bbodata;
    }

        public override
void Init()
{
    this.bboData =
HistoryDataSeries as BBOData;
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
BBOData) && this.bboData.Count
> 1)
    {
        var time =
bbodata.GetTimeUtc(1);

Notification.Print($"Last
tick's time is {time}");
}
```

```
        }  
    }  
}
```

See Also

[Reference](#)

[BBOData Class](#)

[TradeApi.History Namespace](#)

BBODataGetVolume Method

The volume of the required Historical tick

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetVolume(Copy
                        int offset = 0
                    )
```

Parameters

offset [Int32](#) (Optional)
bbo's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
```

```
    public class
GetVolumeExample : 
StrategyBuilder
{
    public
GetVolumeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetVolumeExample";
    #endregion
}
    public BBOData
bbodata;

    public override
void Init()
{
    this.bboData =
HistoryDataSeries as BBOData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
BBOData) && this.bboData.Count
> 1)
    {
        var volume
= bboData.GetVolume(1);

Notification.Print($"Last
tick's volume is {volume}");
    }
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[BBOData Class](#)

[TradeApi.History Namespace](#)

BuildFrom Enumeration

Type of the source data from which is build

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum BuildFrom
```

[Copy](#)

► Members

Member name	Value	Description
Ticks	0	Ticks
Second	1	Second
Minute	2	Minute
Hour	3	Hour
Day	4	Day
Week	5	Week
Month	6	Month
Year	7	Year

See Also

Reference

[TradeApi.History Namespace](#)

ClusterDataType Enumeration

Type of the ClusterData

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum ClusterDataType
```

[Copy](#)

► Members

Member name	Value	Description
Delta	0	Delta
DeltaPercent	1	DeltaPercent
DeltaVolume	1	DeltaVolume
Volume	2	Volume
BuyVolume	3	BuyVolume
SellVolume	4	SellVolume
Trades	11	Trades
MaxTickVolume	15	MaxTickVolume

VolumePercent	16	VolumePercent
CustomPair	18	CustomPair
VolumeImbalance	20	VolumeImbalance

◀ See Also

Reference

[TradeApi.History Namespace](#)

ClusterHistoricalRequest Class

Cluster data based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryClusterHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
ClusterHistoricalRequest :  
HistoricalRequest
```

[Copy](#)

The [ClusterHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 ClusterHistoricalRequest	Cluster data based historical request

[Top](#)

Properties

Name	Description
 BuildFrom	Represents the type of build from period
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 Value	Represent the value of historical elements

[Top](#)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

Reference

[TradeApi.History Namespace](#)

ClusterHistorical Request Constructor

Cluster data based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

[Copy](#)

```
public  
ClusterHistoricalRequest(  
    Instrument instrument,  
    DataType dataType,  
    BuildFrom buildFrom,  
    int value,  
    ClusterDataType  
    clusterDataType  
)
```

Parameters

instrument [Instrument](#)

instrument based on which request will
be processed

dataType [DataType](#)

data type based on which request will
be processed

buildFrom [BuildFrom](#)

represents period type based on which
cluster will be generated

value [Int32](#)

value of historical elements based on
which request will be processed
clusterDataType [ClusterDataType](#)
the cluster data based on which request
will be processed

Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ClusterHistoricalRequestExempl
e : StrategyBuilder
    {
        public
ClusterHistoricalRequestExempl
e()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"ClusterHistoricalRequestExamp
le";
            #endregion
        }

        public override
void Init()
        {
            var
```

Copy

```
HistoricalReq = new
ClusterHistoricalRequest(instr
ument:
InstrumentsManager.Current,
dataType: DataType.Trade,

buildFrom: BuildFrom.Ticks,
value: 1, clusterDataType:
ClusterDataType.Delta);
var interval =
new
Interval(DateTime.UtcNow.AddDa
ys(-2), DateTime.UtcNow);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, interval);
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
if
(loadedHistoricalData != null)
{
for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
var
```

```
    high =
        loadedHistoricalData.GetValue(
            PriceType.High, i);
                var low
=
    loadedHistoricalData.GetValue(
        PriceType.Low, i);
                var
close =
    loadedHistoricalData.GetValue(
        PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[ClusterHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ClusterHistorical Request Properties

The [ClusterHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 BuildFrom	Represents the type of build from period
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes)

(Inherited from
[HistoricalRequest](#))



[Value](#)

Represent the
value of historical
elements

[Top](#)

◀ See Also

[Reference](#)

[ClusterHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ClusterHistoricalRequestBuildFromProperty

Represents the type of build from period

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuildFrom BuildFrom {  
    get; }
```

[Copy](#)

Property Value

[BuildFrom](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;
```

[Copy](#)

```
namespace TestEnv
```

```
{
```

```
    public class
```

```
BuildFromExample :
```

```
StrategyBuilder
{
    public
BuildFromExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BuildFromExample";
    #endregion
}

        public override
void Init()
{
    BuildFrom?
period = null;

if(HistoryDataSeries.Historica
lRequest is
ClusterHistoricalRequest)
    period =
(HistoryDataSeries.HistoricalR
equest as
ClusterHistoricalRequest)?.Bui
ldFrom;
    var
HistoricalReq = new
ClusterHistoricalRequest(instru
ment:
InstrumentsManager.Current,
dataType: DataType.Trade,

buildFrom: period ??
BuildFrom.Ticks, value: 1,
clusterDataType:
ClusterDataType.Delta);
    var interval =
```

```
new  
Interval(DateTime.UtcNow.AddDays(-2), DateTime.UtcNow);  
  
HistoricalDataManager.OnLoaded  
+=  
this.HistoricalDataManager_OnL  
oaded;  
  
HistoricalDataManager.Get(Hist  
oricalReq, interval);  
}  
  
private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
            var low  
=  
loadedHistoricalData.GetValue(  
PriceType.Low, i);  
            var  
close =  
loadedHistoricalData.GetValue(
```

```
    PriceType.Close, i);

    Notification.Print(string.Format
        ("Open: {0}, High: {1}, Low:
        {2}, Close: {3}", open, high,
        low, close));
    }
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[ClusterHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ClusterHistorical RequestValue Property

Represent the value of historical elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; }
```

[Copy](#)

Property Value
[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
    {
        public
ValueExample()
```

[Copy](#)

```
        : base()
    {
        #region Initialization

        Credentials.ProjectName =
"ValueExample";
        #endregion
    }

        public override
void Init()
{
    int? value =
null;
    if
(HistoryDataSeries.HistoricalR
equest is
ClusterHistoricalRequest)
        value =
(HistoryDataSeries.HistoricalR
equest as
ClusterHistoricalRequest)?.Val
ue;
    var
HistoricalReq = new
ClusterHistoricalRequest(instr
ument:
InstrumentsManager.Current,
dataType: DataType.Trade,

buildFrom: BuildFrom.Ticks,
value: value ?? 1,
clusterDataType:
ClusterDataType.Delta);
    var interval =
new
Interval(DateTime.UtcNow.AddDa
ys(-2), DateTime.UtcNow);
```

```
    HistoricalDataManager.OnLoaded
    +=
    this.HistoricalDataManager_OnL
    oaded;

    HistoricalDataManager.Get (Hist
    oricalReq, interval);
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Form
at("Open: {0}, High: {1}, Low: {2}"), open, high, low);
}
}
```

```
 {2}, Close: {3}"}, open, high,  
 low, close));  
 }  
 }  
  
 public override  
 void Update(TickStatus args)  
 {  
 }  
 }  
 }
```

See Also

[Reference](#)

[ClusterHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ClusterHistorical Request Methods

The [ClusterHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[ClusterHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

CustomData Class

Custom data entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalData](#)
[TradeApi.HistoryCustomData](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public sealed class CustomData :  
    HistoricalData,  
    IDisposable
```

Copy

The [CustomData](#) type exposes the following members.

▪ Constructors

	Name	Description
 	CustomData	Sets the custom data series

[Top](#)

▪ Properties

	Name	Description
 	Count	The count of the custom data elements (Overrides HistoricalDataCount)
 	HistoricalRequest	A collection of the unit indexes which are the lowest values within interval

(Overrides
[HistoricalDataHistoricalRequest](#))



Item

Returns value from custom historian series by index

[Top](#)

Methods

	Name	Description
	Dispose	Releases all resources used by the CustomData
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	Finalize	(Overrides ObjectFinalize)
	FindInterval	Index of the historical unit by given time in UTC (Inherited from HistoricalData)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetHighest	A collection of the unit indexes which are the highest values within interval (Overrides HistoricalDataGetHighest(PriceType, Interval))
	GetLowest	A collection of the unit indexes which are the lowest values within interval (Overrides HistoricalDataGetLowest(PriceType, Interval))
	GetTimeUtc	Returns the time of custom

elements being added in UTC
(Overrides
[HistoricalDataGetTimeUtc\(Int32\)](#))

 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	GetValue	Returns the value from custom data series (Overrides HistoricalDataGetValue(PriceType, Int32))
 	SetValue	Sets the value to custom data series
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

Reference

[TradeApi.History Namespace](#)

CustomData Constructor

Sets the custom data series

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public CustomData(  
    HistoricalData  
    historicalData  
)
```

[Copy](#)

Parameters

historicalData [HistoricalData](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
CustomDataExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
CustomDataExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"CustomDataExample";
    #endregion
}

    CustomData
customData;

    public override
void Init()
{
    customData =
new
CustomData(HistoryDataSeries);

HistoricalDataManager.AddCusto
m("myDATA", customData);
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{

customData.SetValue(PriceType.
Close, 123);
}
}
```

```
        }  
    }
```

See Also

Reference

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomData Properties

The [CustomData](#) type exposes the following members.

► Properties

	Name	Description
 	Count	The count of the custom data elements (Overrides HistoricalDataCount)
 	HistoricalRequest	A collection of the unit indexes which are the lowest values within interval (Overrides HistoricalDataHistoricalRequest)
 	Item	Returns value from custom historian series by index

[Top](#)

► See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataCount Property

The count of the custom data elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override int Count {  
    get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
CountExample : StrategyBuilder  
    {  
        public  
CountExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"CountExample";
            #endregion
        }

        CustomData
customData;

        public override
void Init()
{
    customData =
new
CustomData(HistoryDataSeries);

HistoricalDataManager.AddCusto
m("myDATA", customData);
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var count =
customData.Count;

Notification.Print($"Custom
unit count is {count}");
    }
}
}
```

See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataHistoricalRequest Property

A collection of the unit indexes which are the lowest values within interval

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override  
HistoricalRequest  
HistoricalRequest { get; }
```

[Copy](#)

Property Value

[HistoricalRequest](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
GetLowestExample :  
StrategyBuilder  
{
```

```
        public
GetLowestExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetLowestExample";
    #endregion
}

        CustomData
customData;

        public override
void Init()
{
    customData =
new
CustomData(HistoryDataSeries);

HistoricalDataManager.AddCusto
m("myDATA", customData);
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{

customData.SetValue(PriceType.
Close, 123);
}
}
}
```

See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataItem Property

Returns value from custom historian series by index

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double this[Copy
    PriceType priceType,
    int offset = 0
] { get; set; }
```

Parameters

priceType [PriceType](#)

a series where value is stored

offset [Int32](#) (Optional)

The element offset in the custom series

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Indicators;
```

```
using TradeApi.History;

namespace TestEnv
{
    public class
IndexerExample :
IndicatorBuilder
{
    public
IndexerExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"IndexerExample";
        #endregion

Lines.Set("IndexerExample");

base.SeparateWindow = false;
    }

    CustomData
customData;

    public override
void Init()
    {
        customData =
new
CustomData(HistoryDataSeries);

HistoricalDataManager.AddCusto
m("myDATA", customData);
    }
}
```

```
        public override
void Update(TickStatus args)
{
    //setting

customData[TradeApi.PriceType.
Close, 1] = 123;
    //getting
    var valGet =
customData[TradeApi.PriceType.
Close, 1];
}
}
```

See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomData Methods

The [CustomData](#) type exposes the following members.

▲ Methods

Name	Description
 Dispose	
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	(Overrides ObjectFinalize)
  FindInterval	Index of the historical unit by given time in UTC (Inherited from HistoricalData)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
  GetHighest	A collection of the unit indexes which are the highest values within interval (Overrides HistoricalDataGetHighest(PriceType, Interval))
  GetLowest	A collection of the unit indexes which are the lowest values within interval (Overrides HistoricalDataGetLowest(PriceType, Interval))
  GetTimeUtc	Returns the time of custom elements being added in UTC

(Overrides
[HistoricalDataGetTimeUtc\(Int32\)](#))

 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	GetValue	Returns the value from custom data series (Overrides HistoricalDataGetValue(PriceType, Int32))
 	SetValue	Sets the value to custom data series
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

▲ See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataDispose Method

Releases all resources used by the
[CustomData](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public void Dispose()
```

[Copy](#)

Implements
[IDisposableDispose](#)

◀ See Also

Reference

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataFinalize Method

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
protected override void  
Finalize()
```

[Copy](#)

Implements
[ObjectFinalize](#)

► See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataGet Highest Method

A collection of the unit indexes which are the highest values within interval

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override List<int>
GetHighest(
    PriceType type,
    Interval interval
)
```

[Copy](#)

Parameters

type [PriceType](#)

interval [Interval](#)

Return Value

[ListInt32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System;
```

[Copy](#)

```
using System.Linq;
using TradeApi.History;

namespace TestEnv
{
    public class
GetHighestExample :
StrategyBuilder
{
    public
GetHighestExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetHighestExample";
        #endregion
    }

    CustomData
customData;

    public override
void Init()
    {
        customData =
new
CustomData(HistoryDataSeries);

HistoricalDataManager.AddCusto
m("myDATA", customData);
    }

    public override
void Update(TickStatus args)
    {
        if (args ==
TickStatus.IsBar)
```

```
        {
            var
highestArr =
customData.GetHighest(PriceTyp
e.Close, new
Interval(DateTime.UtcNow,
DateTime.UtcNow.AddDays(-3)));
Notification.Print($"the
highest close within last
three day's is
{highestArr.FirstOrDefault()}");
}
}
}
```

See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataGet Lowest Method

A collection of the unit indexes which are the lowest values within interval

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override List<int>
GetLowest(
    PriceType type,
    Interval interval
)
```

[Copy](#)

Parameters

type [PriceType](#)

interval [Interval](#)

Return Value

[ListInt32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
```

[Copy](#)

```
using System;
using System.Linq;

namespace TestEnv
{
    public class
GetLowestExample : 
StrategyBuilder
{
    public
GetLowestExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetLowestExample";
        #endregion
    }

    CustomData
customData;

    public override
void Init()
{
    customData =
new
CustomData(HistoryDataSeries);

HistoricalDataManager.AddCusto
m("myDATA", customData);
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
```

```
        {
            var
lowestArr =
customData.GetLowest(PriceType
.Close, new
Interval(DateTime.UtcNow,
DateTime.UtcNow.AddDays(-3)));
Notification.Print($"the
lowest close within last three
day's is
{lowestArr.FirstOrDefault()}");
;
        }
    }
}
```

See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataGetTime Utc Method

Returns the time of custom elements being added in UTC

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public override DateTime  
GetTimeUtc(  
    int offset = 0  
)  
Copy
```

Parameters

offset [Int32](#) (Optional)
unit's index offset

Return Value
[DateTime](#)

◀ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System;  
Copy
```

```
namespace TestEnv
{
    public class GetTimeUtcExample : StrategyBuilder
    {
        public GetTimeUtcExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "GetTimeUtcExample";
            #endregion
        }

        CustomData
        customData;

        public override
        void Init()
        {
            customData =
                new
                CustomData(HistoryDataSeries);

            HistoricalDataManager.AddCusto
            m("myDATA", customData);
        }

        public override
        void Update(TickStatus args)
        {

            if(customData.GetTimeUtc() >
            DateTime.UtcNow)
        }
}
```

```
        customData.SetValue(PriceType.  
Close, 123);  
    }  
}  
}
```

See Also

Reference

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataGetValue Method

Returns the value from custom data series

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override double  
GetValue(  
    PriceType priceType,  
    int offset = 0  
)
```

[Copy](#)

Parameters

priceType [PriceType](#)

the price type of the custom data

offset [Int32](#) (Optional)

unit's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;
```

[Copy](#)

```
using TradeApi.History;

namespace TestEnv
{
    public class
GetValueExample :
StrategyBuilder
{
    public
GetValueExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetValueExample";
        #endregion
    }

    CustomData
customData;

    public override
void Init()
    {
        customData =
new
CustomData(HistoryDataSeries);

HistoricalDataManager.AddCusto
m("myDATA", customData);
    }

    public override
void Update(TickStatus args)
    {

customData.SetValue(PriceType.
Close, 123);
```

```
        if (args ==  
    TickStatus.IsBar)  
        {  
            var close =  
    customData.GetValue(PriceType.  
Close, 1);  
  
Notification.Print($"the last  
custom bar's close is  
{close}");  
        }  
    }  
}
```

See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

CustomDataSetValue Method

Sets the value to custom data series

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void SetValue(Copy
    PriceType priceType,
    double value,
    int offset = 0
)
```

Parameters

priceType [PriceType](#)

a series where value is stored

value [Double](#)

value to be set to custom data series

offset [Int32](#) (Optional)

The element offset in the custom data series

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
```

Copy

```
using TradeApi.History;

namespace TestEnv
{
    public class SetValueExample : StrategyBuilder
    {
        public SetValueExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "SetValueExample";
            #endregion
        }

        CustomData
        customData;

        public override
        void Init()
        {
            customData =
                new
                CustomData(HistoryDataSeries);

            HistoricalDataManager.AddCustom(
                "myDATA", customData);
        }

        public override
        void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar)
            {

```

```
        customData.SetValue(PriceType.  
Close, 123);  
    }  
}  
}
```

See Also

[Reference](#)

[CustomData Class](#)

[TradeApi.History Namespace](#)

DataType Enumeration

Supported history data types

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum DataType
```

[Copy](#)

► Members

Member name	Value	Description
Bid	1	Bid type
Trade	4	Trade type
Ask	1,000	Ask type

► See Also

Reference

[TradeApi.History Namespace](#)

HeikinAshiHistoricalRequest Class

HeikinAshi based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryHeikinAshiHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class  
HeikinAshiHistoricalRequest :  
HistoricalRequest
```

[Copy](#)

The [HeikinAshiHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 HeikinAshiHistoricalRequest	HeikinAshi based historical request

[Top](#)

Properties

Name	Description
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 Period	Represents period type based on which Heikin Ashi will be processed
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 Value	Represent the amount of historical elements

[Top](#)

Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
  GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ **See Also**

[Reference](#)

[TradeApi.History Namespace](#)

HeikinAshiHistorical Request Constructor

HeikinAshi based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public  
HeikinAshiHistoricalRequest(  
    Instrument instrument,  
    DataType dataType,  
    Period period,  
    int value  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

instrument based on which request will
be processed

dataType [DataType](#)

data type based on which request will
be processed

period [Period](#)

represents period type based on which
Heikin Ashi will be processed

value [Int32](#)

represent the amount of historical
elements

Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
HeikinAshiHistoricalRequestExa
mple : StrategyBuilder
{
    public
HeikinAshiHistoricalRequestExa
mple()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"HeikinAshiHistoricalRequestEx
ample";
        #endregion
    }

    public override
void Init()
{
    var
newHistoricalReq = new
HeikinAshiHistoricalRequest(in
strument:
InstrumentsManager.Current,
dataType: DataType.Trade,
period: Period.Hour, value:
15);
```

Copy

```
        var interval =
    new
Interval(DateTime.UtcNow.AddDa
ys(-2), DateTime.UtcNow);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(newH
istoricalReq, interval);
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
    {
        for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
        {
            var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
            var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
            var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
            var
close =
```

```
loadedHistoricalData.GetValue(PriceType.Close, i);

Notification.Print(string.Format("Open: {0}, High: {1}, Low: {2}, Close: {3}", open, high, low, close));
}

}

public override void Update(TickStatus args)
{
}

}
```

See Also

Reference

HeikinAshiHistoricalRequest Class

TradeApi.History Namespace

HeikinAshiHistorical Request Properties

The [HeikinAshiHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 Period	Represents period type based on which Heikin Ashi will be processed
 SubscribeToQuotes	Subscribe to quotes (history will be change on

quotes)
(Inherited from
[HistoricalRequest](#))



[Value](#)

Represent the amount of historical elements

[Top](#)

◀ See Also

[Reference](#)

[HeikinAshiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HeikinAshiHistoricalRequestPeriod Property

Represents period type based on which Heikin Ashi will be processed

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Period Period { get; } Copy
```

Property Value

[Period](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
PeriodExample :
StrategyBuilder
```

```
    {
        public
PeriodExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"PeriodExample";
            #endregion
        }

        public override
void Init()
{
    Period? period
= null;
    if
(HistoryDataSeries.HistoricalR
equest is
HeikinAshiHistoricalRequest)
        period =
(HistoryDataSeries.HistoricalR
equest as
HeikinAshiHistoricalRequest)?.
Period;
    var
HistoricalReq = new
HeikinAshiHistoricalRequest(in
strument:
InstrumentsManager.Current,
dataType: DataType.Trade,
period: period ?? Period.Hour,
value: 15);
    var interval =
new
Interval(DateTime.UtcNow.AddDa
ys(-2), DateTime.UtcNow);
```

```
    HistoricalDataManager.OnLoaded
    +=
    this.HistoricalDataManager_OnL
    oaded;

    HistoricalDataManager.Get (Hist
    oricalReq, interval);
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Form
at("Open: {0}, High: {1}, Low: {2}"), open, high, low);
}
}
```

```
 {2}, Close: {3}"}, open, high,  
 low, close));  
 }  
 }  
  
 public override  
 void Update(TickStatus args)  
 {  
 }  
 }  
 }
```

See Also

Reference

[HeikinAshiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HeikinAshiHistorical RequestValue Property

Represent the amount of historical elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; }
```

[Copy](#)

Property Value
[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
    {
        public
ValueExample()
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"ValueExample";
            #endregion
        }

    public override
void Init()
{
    int? value =
null;
    if
(HistoryDataSeries.HistoricalR
equest is
HeikinAshiHistoricalRequest)
        value =
(HistoryDataSeries.HistoricalR
equest as
HeikinAshiHistoricalRequest)?.Value;
    var
HistoricalReq = new
HeikinAshiHistoricalRequest(in
strument:
InstrumentsManager.Current,
dataType: DataType.Trade,
period: Period.Hour, value:
value ?? 1);
    var interval =
new
Interval(DateTime.UtcNow.AddDa
ys(-2), DateTime.UtcNow);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
```

```
HistoricalDataManager.Get(Hist  
oricalReq, interval);  
}  
  
        private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
            var low  
=  
loadedHistoricalData.GetValue(  
PriceType.Low, i);  
            var  
close =  
loadedHistoricalData.GetValue(  
PriceType.Close, i);  
  
Notification.Print(string.Format  
("Open: {0}, High: {1}, Low:  
{2}, Close: {3}", open, high,  
low, close));  
    }  
}
```

```
        }

    public override
void Update(TickStatus args)
{
    }

}
```

See Also

Reference

[HeikinAshiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HeikinAshiHistorical Request Methods

The [HeikinAshiHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[HeikinAshiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HistoricalData Class

Historical data entity

► Inheritance Hierarchy

```
SystemObject  TradeApi.HistoryHistoricalData
              TradeApi.HistoryBarData
              TradeApi.HistoryBBOData
              TradeApi.HistoryCustomData
              TradeApi.HistoryTradeData
```

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public abstract class
HistoricalData
```

[Copy](#)

The `HistoricalData` type exposes the following members.

► Properties

	Name	Description
	Count	The count of the historical data elements
	HistoricalRequest	Reference to the

initial
HistoricalRequest
of current
HistoricalData

[Top](#)

◀ Methods

Name	Description
 Check	
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
 FindInterval	Index of the

		historical unit by given time in UTC
	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetHighest	A collection of the unit indexes which are the highest values within interval
 	GetLowest	A collection of the unit indexes which are the lowest values within interval
 	GetTimeUtc	Retrieves historical value by index based on the price type
	GetType	Gets the Type of the

current
instance.
(Inherited
from [Object](#))



[GetValue](#)

Retrieves
historical
value by
index based
on price
type



[MemberwiseClone](#)

Creates a
shallow copy
of the
current
[Object](#).
(Inherited
from [Object](#))



[ToString](#)

Returns a
string that
represents
the current
object.
(Inherited
from [Object](#))

[Top](#)

◀ Fields

	Name	Description
	cache	
	cashItem	

[Top](#)

See Also

Reference

[TradeApi.History Namespace](#)

HistoricalData Properties

The [HistoricalData](#) type exposes the following members.

Properties

	Name	Description
 	Count	The count of the historical data elements
 	HistoricalRequest	Reference to the initial HistoricalRequest of current HistoricalData

[Top](#)

See Also

[Reference](#)

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDataCount Property

The count of the historical data elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public virtual int Count {  
    get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
CountExample : StrategyBuilder  
    {  
        public  
CountExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"CountExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var count =
this.HistoryDataSeries.Count;

Notification.Print($"Historical bar's count is {count}");
    }
}
}
```

See Also

Reference

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDataHistoricalRequest Property

Reference to the initial HistoricalRequest of current HistoricalData

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public virtual  
HistoricalRequest  
HistoricalRequest { get; }
```

[Copy](#)

Property Value

[HistoricalRequest](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
HistoricalRequestExample :  
StrategyBuilder  
{  
    public  
HistoricalRequestExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"HistoricalRequestExample";  
        #endregion  
    }  
  
    public override  
void Init()  
    {  
    }  
  
    public override  
void Update(TickStatus args)  
    {  
        if (args ==  
TickStatus.IsBar)  
        {  
            var  
instrument =  
this.HistoryDataSeries.Histori  
calRequest.Instrument;  
  
Notification.Print($"the  
historical request  
instrument's symbol is  
{instrument.Symbol}");  
        }  
    }  
}
```

See Also

[Reference](#)

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalData Methods

The [HistoricalData](#) type exposes the following members.

▲ Methods

	Name	Description
	Check	
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection.

		(Inherited from Object)
 	FindInterval	Index of the historical unit by given time in UTC
 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetHighest	A collection of the unit indexes which are the highest values within interval
 	GetLowest	A collection of the unit indexes which are the lowest values within interval
 	GetTimeUtc	Retrieves historical value by index based

on the price type

 GetType	Gets the Type of the current instance. (Inherited from Object)
  GetValue	Retrieves historical value by index based on price type
 MemberwiseClone	Creates a shallow copy of the current Object . (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

[HistoricalData Class](#)

TradeApi.History Namespace

HistoricalDataCheck Method

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
protected void Check(  
    int offset  
)
```

[Copy](#)

Parameters

offset [Int32](#)

◀ See Also

Reference

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDataFind Interval Method

Index of the historical unit by given time in UTC

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int FindInterval(Copy
    DateTime timeUtc
)
```

Parameters

timeUtc [DateTime](#)

the DateTime of the searched interval

Return Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System.Linq;
using System;
```

```
namespace TestEnv
{
    public class
FindIntervalExample : 
StrategyBuilder
{
    public
FindIntervalExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"FindIntervalExample";
        #endregion
    }

    public override
void Init()
    {
    }

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
interval =
this.HistoryDataSeries.FindInt
erval(DateTime.UtcNow.AddDays(
-3));
Notification.Print($"the unit
index is {interval}");
    }
}
```

```
        }  
    }
```

See Also

Reference

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDataGet Highest Method

A collection of the unit indexes which are the highest values within interval

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public virtual List<int> GetHighest(Copy
    PriceType type,
    Interval interval
)
```

Parameters

type [PriceType](#)

the price type of the highest search

interval [Interval](#)

the DateTime interval to be filled in collection

Return Value

[ListInt32](#)

◀ Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System.Linq;
using System;

namespace TestEnv
{
    public class
GetHighestExample :
StrategyBuilder
{
    public
GetHighestExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetHighestExample";
        #endregion
    }

    public override
void Init()
    {
    }

    public override
void Update(TickStatus args)
    {
        if (args ==
TickStatus.IsBar)
        {
            var
highestArr =
this.HistoryDataSeries.GetHigh
est(PriceType.Close, new
Interval(DateTime.UtcNow,
```

```
DateTime.UtcNow.AddDays(-3))));

Notification.Print($"the
highest close within last
three day's is
{highestArr.FirstOrDefault()}");
);

}

}
```

See Also

[Reference](#)

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDataGet Lowest Method

A collection of the unit indexes which are the lowest values within interval

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public virtual List<int>
GetLowest(
    PriceType type,
    Interval interval
)
```

[Copy](#)

Parameters

type [PriceType](#)

the price type of the lowest search

interval [Interval](#)

the DateTime interval to be filled in collection

Return Value

[ListInt32](#)

◀ Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System.Linq;
using System;

namespace TestEnv
{
    public class
GetLowestExample :
StrategyBuilder
{
    public
GetLowestExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetLowestExample";
        #endregion
    }

    public override
void Init()
    {
    }

    public override
void Update(TickStatus args)
    {
        if (args ==
TickStatus.IsBar)
        {
            var
lowestArr =
this.HistoryDataSeries.GetLowe
st(PriceType.Close, new
Interval(DateTime.UtcNow,
```

```
DateTime.UtcNow.AddDays(-3))));

Notification.Print($"the
lowest close within last three
day's is
{lowestArr.FirstOrDefault()}");
;

}
}

}
```

See Also

[Reference](#)

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDataGetTime Utc Method

Retrieves historical value by index based on the price type

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public abstract DateTime  
GetTimeUtc(  
            int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
unit's index offset

Return Value
[DateTime](#)

◀ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System.Linq;
```

[Copy](#)

```
namespace TestEnv
{
    public class GetTimeUtcexample : StrategyBuilder
    {
        public GetTimeUtcexample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "GetTimeUtcexample";
            #endregion
        }

        public override void Init()
        {
        }

        public override void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar)
            {
                var time =
                    this.HistoryDataSeries.GetTime
                    Utc(1);

                Notification.Print($"the time
                    of the last unit is {time}");
            }
        }
    }
}
```

See Also

Reference

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDataGet Value Method

Retrieves historical value by index based on price type

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public virtual double  
GetValue(  
    PriceType priceType,  
    int offset = 0  
)
```

[Copy](#)

Parameters

priceType [PriceType](#)

the price type of the data

offset [Int32](#) (Optional)

the DateTime interval to be filled in collection

Return Value

[Double](#)

◀ Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System.Linq;

namespace TestEnv
{
    public class
GetValueExample :
StrategyBuilder
{
    public
GetValueExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetValueExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var close =
this.HistoryDataSeries.GetValue(PriceType.Close, 1);

Notification.Print($"the last
bar's close is {close}");
    }
}
```

```
        }  
    }  
}
```

See Also

[Reference](#)

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalData Fields

The [HistoricalData](#) type exposes the following members.

► Fields

	Name	Description
	cache	
	cashItem	

[Top](#)

► See Also

[Reference](#)

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDatacache Field

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
protected [T:C8k=.Csk=] cache Copy
```

Field Value

[T:C8k=.Csk=]

◀ See Also

[Reference](#)

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDatacash Item Field

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
protected [T:dM4=.qM4=]  
cashItem
```

[Copy](#)

Field Value

[T:dM4=.qM4=]

◀ See Also

[Reference](#)

[HistoricalData Class](#)

[TradeApi.History Namespace](#)

HistoricalDataManager Class

Historical data manager entity

► Inheritance Hierarchy

[SystemObject](#) `TradeApi.HistoryHistoricalDataManager`

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
HistoricalDataManager
```

[Copy](#)

The `HistoricalDataManager` type exposes the following members.

► Methods

	Name	Description
	AddCustom	Adds custom data with synchronized internal requested data

 	ChangeInterval	Updates historian interval of the series
 	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 	Get	Adds access to historical data based on requested type
 	GetCustom	Gets access to custom data with synchronized internal requested data
 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetType	Gets the Type of the current instance.

		(Inherited from Object)
 	Remove	Removes HistoricalData from the cache
 	RemoveCustom	Removes custom historical data from the cache
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

Fields

	Name	Description
 	OnLoaded	Occurs when the historical data was loaded
 	OnRemoved	Occurs when the historical data was removed

[Top](#)

See Also

Reference

[TradeApi.History Namespace](#)

HistoricalDataManager Methods

The [HistoricalDataManager](#) type exposes the following members.

▲ Methods

	Name	Description
 	AddCustom	Adds custom data with synchronized internal requested data
 	ChangeInterval	Updates historian interval of the series
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 	Get	Adds access to historical data based

		on requested type
	GetCustom	Gets access to custom data with synchronized internal requested data
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	Remove	Removes HistoricalData from the cache
	RemoveCustom	Removes custom historical data from the cache
	ToString	Returns a string that represents the current object.

(Inherited
from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalData ManagerAddCustom Method

Adds custom data with synchronized internal requested data

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void AddCustom(Copy
    string name,
    CustomData history
)
```

Parameters

name [String](#)

the name of the custom historical data

history [CustomData](#)

the custom historical data

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
```

Copy

```
namespace TestEnv
{
    public class AddCustomExample : StrategyBuilder
    {
        public AddCustomExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "AddCustomExample";
            #endregion
        }

        CustomData
        customData;

        public override
        void Init()
        {
            customData =
                new
                CustomData(HistoryDataSeries);

            HistoricalDataManager.AddCusto
            m("myDATA", customData);
        }

        public override
        void Update(TickStatus args)
        {
        }
    }
}
```

See Also

Reference

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalData ManagerChange Interval Method

Updates historian interval of the series

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void ChangeInterval(Copy
    HistoricalData
    historicalData,
    Interval interval
)
```

Parameters

historicalData [HistoricalData](#)

fetched historical data

interval [Interval](#)

new interval with depth of the historical
data

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
```

[Copy](#)

```
using TradeApi.History;
using System;
using System.Linq;

namespace TestEnv
{
    public class
ChangeIntervalExample : StrategyBuilder
{
    public
ChangeIntervalExample()
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"ChangeIntervalExample";
        #endregion
    }

    public Interval
interval;
    public
HistoricalData
loadedHistoricalData;

    public override
void Init()
{
    var timeRequest
= new
TimeHistoricalRequest(Instруме
ntsManager.Current,
HistoryDataSeries.HistoricalRe
quest.DataType, Period.Minute,
1);
    interval = new
Interval(DateTime.UtcNow.AddDa
```

```
    ys(-2), DateTime.UtcNow);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(time
Request, interval);
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{

this.loadedHistoricalData =
loadedHistoricalData;
if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
    {
        var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
        var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
close =
```

```
        loadedHistoricalData.GetValue(
            PriceType.Close, i);

        Notification.Print(string.Format
            ("Open: {0}, High: {1}, Low:
            {2}, Close: {3}", open, high,
            low, close));
    }
}

public override
void Update(TickStatus args)
{
    if
        (HistoryDataSeries.GetTimeUtc(
            HistoryDataSeries.Count - 1) <
            tickData.GetTimeUtc(tickData.C
            ount - 1))
    {
        interval =
            new
            Interval(interval.FromUtc.AddDays(-1), DateTime.UtcNow);

        HistoricalDataManager.ChangeIn
        terval(loadedHistoricalData,
            interval);
    }
}
}
```

See Also

Reference

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalData ManagerGet Method

Adds access to historical data based on requested type

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public HistoricalData Get(Copy
    HistoricalRequest
    request,
    Interval interval
)
```

Parameters

request [HistoricalRequest](#)

historical request of the future fetched
historical data

interval [Interval](#)

interval with depth of the historical data

Return Value

[HistoricalData](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System;
using System.Linq;

namespace TestEnv
{
    public class GetExample
    : StrategyBuilder
    {
        public GetExample()
        : base()
        {
            #region
            Initialization

            Credentials.ProjectName =
            "GetExample";
            #endregion
        }

        public override
void Init()
{
    var timeRequest
= new
TimeHistoricalRequest(Instруме
ntsManager.Current,
HistoryDataSeries.HistoricalRe
quest.DataType, Period.Minute,
1);
    var interval =
new
Interval(DateTime.UtcNow.AddDays(-2), DateTime.UtcNow);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
```

```
oaded;

HistoricalDataManager.Get(time
Request, interval);
}
private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
```

```
        }

    public override
void Update(TickStatus args)
{
    }

}
```

See Also

[Reference](#)

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalData ManagerGetCustom Method

Gets access to custom data with synchronized internal requested data

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public CustomData GetCustom(Copy
    string name
)
```

Parameters

name [String](#)

the name of the custom historical data

Return Value

[CustomData](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
```

```
namespace TestEnv
{
    public class GetCustomExample : StrategyBuilder
    {
        public GetCustomExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "GetCustomExample";
            #endregion
        }

        public override void Init()
        {
            var customData
            = new CustomData(HistoryDataSeries);
            HistoricalDataManager.AddCustom("myDATA", customData);
        }

        public override void Update(TickStatus args)
        {
            var customData
            =
            HistoricalDataManager.GetCustom("myDATA");

            customData.SetValue(PriceType.
                Close, 123);
        }
    }
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalData ManagerRemove Method

Removes HistoricalData from the cache

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void Remove(  
    HistoricalData  
    historicalData  
)
```

[Copy](#)

Parameters

historicalData [HistoricalData](#)

fetched historical data to be removed

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System;  
using System.Linq;
```

[Copy](#)

```
namespace TestEnv
{
    public class RemoveExample : StrategyBuilder
    {
        public RemoveExample()
            : base()
        {
            #region Initialization

            Credentials.ProjectName =
"RemoveExample";
            #endregion
        }

        private HistoricalData historicalData;
        private bool isLoaded = false;

        public override void Init()
        {
            var timeRequest
= new TimeHistoricalRequest(InstumentsManager.Current,
HistoryDataSeries.HistoricalRequest.DataType, Period.Minute,
1);
            interval = new Interval(DateTime.UtcNow.AddDays(-2), DateTime.UtcNow);

            HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
```

```
oaded;

HistoricalDataManager.OnRemove
d +=
this.HistoricalDataManager_OnR
emoved;

HistoricalDataManager.Get(time
Request, interval);
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
historicalData)
{

this.historicalData =
historicalData;
this.isLoaded
= true;

Notification.Print($"History
data is loaded");
}

private void
HistoricalDataManager_OnRemove
d(HistoricalData
historicalData)
{

this.historicalData = null;
this.isLoaded
= false;

Notification.Print($"History
data is removed");
}
```

```
        public override
void Update(TickStatus args)
{
    if
(this.isLoaded &&
HistoryDataSeries.GetTimeUtc(H
istoryDataSeries.Count - 1) <
tickData.GetTimeUtc(historical
Data.Count - 1))
    {

HistoricalDataManager.Remove(h
istoricalData);
    }
}
}
```



See Also

[Reference](#)

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalData ManagerRemove Custom Method

Removes custom historical data from the cache

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void RemoveCustom(Copy
    string name
)
```

Parameters

name [String](#)

the name of the custom historical data

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{Copy
```

```
    public class
RemoveCustomExample :
StrategyBuilder
{
    public
RemoveCustomExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"RemoveCustomExample";
    #endregion
}

    CustomData
customData;

    public override
void Init()
{
    customData =
new
CustomData(HistoryDataSeries);

HistoricalDataManager.AddCusto
m("myDATA", customData);
}

    public override
void Update(TickStatus args)
{

}

    public override
void Complete()
{
```

```
    HistoricalDataManager.RemoveCu  
    stom("myDATA");  
}  
}  
}
```

See Also

Reference

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalDataManager Fields

The [HistoricalDataManager](#) type exposes the following members.

▲ Fields

	Name	Description
 	OnLoaded	Occurs when the historical data was loaded
 	OnRemoved	Occurs when the historical data was removed

[Top](#)

▲ See Also

[Reference](#)

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalData ManagerOnLoaded Field

Occurs when the historical data was loaded

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Action<HistoricalData>  
OnLoaded
```

[Copy](#)

Field Value

[ActionHistoricalData](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System;  
using System.Linq;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
OnLoadedExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
OnLoadedExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"OnLoadedExample";
    #endregion
}

        public override
void Init()
{
    var timeRequest
= new
TimeHistoricalRequest(Instrome
ntsManager.Current,
HistoryDataSeries.HistoricalRe
quest.DataType, Period.Minute,
1);
    var interval =
new
Interval(DateTime.UtcNow.AddDays(-2), DateTime.UtcNow);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(time
Request, interval);
}

        private void
HistoricalDataManager_OnLoaded
```

```
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var
low =
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}

public override
void Update(TickStatus args)
{
}
```

```
        }  
    }
```

See Also

Reference

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalData ManagerOnRemoved Field

Occurs when the historical data was removed

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Action<HistoricalData>  
OnRemoved
```

[Copy](#)

Field Value

[ActionHistoricalData](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
OnRemovedExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
OnRemovedExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"OnRemovedExample";
        #endregion
    }

        public override void
Init()
{
    var timeRequest =
new
TimeHistoricalRequest(Instруме
ntsManager.Current,
HistoryDataSeries.HistoricalRe
quest.DataType, Period.Minute,
1);
    var interval = new
Interval(DateTime.UtcNow.AddDa
ys(-2), DateTime.UtcNow);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.OnRemove
d +=
this.HistoricalDataManager_OnR
emoved;

HistoricalDataManager.Get(time
Request, interval);
```

```
    }

        private void
HistoricalDataManager_OnLoaded
(HistoricalData
historicalData)
{
}

Notification.Print($"History
data is loaded");

HistoricalDataManager.Remove(h
istoricalData);
}
private void
HistoricalDataManager_OnRemove
d(HistoricalData
historicalData)
{
}

Notification.Print($"History
data is removed");
}

public override void
Update(TickStatus args)
{
}
}
```

See Also

Reference

[HistoricalDataManager Class](#)

[TradeApi.History Namespace](#)

HistoricalRequest Class

Historical Request

▪ Inheritance Hierarchy

[SystemObject](#) `TradeApi.HistoryHistoricalRequest`

[More](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public abstract class  
    HistoricalRequest
```

[Copy](#)

The `HistoricalRequest` type exposes the following members.

▪ Constructors

Name	Description
 HistoricalRequest	Historical Request

[Top](#)

▪ Properties

Name	Description
 DataType	Represent the history data type
 Instrument	Represents the Instrument for selected historical data
 LoadExtendedSession	Load Extended Session history
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes)

[Top](#)

Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)

	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MemberwiseClone	Creates a

shallow copy
of the
current
[Object](#).
(Inherited
from [Object](#))



[ToString](#)

Returns a
string that
represents
the current
object.
(Inherited
from [Object](#))

[Top](#)

▪ See Also

[Reference](#)

[TradeApi.History Namespace](#)

▪ Inheritance Hierarchy

[SystemObject](#)

[TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryClusterHistoricalRequest](#)

[TradeApi.HistoryHeikinAshiHistoricalRequest](#)

[TradeApi.HistoryKagiHistoricalRequest](#)

[TradeApi.HistoryLineBreakHistoricalRequest](#)

[TradeApi.HistoryPointAndFigureHistoricalRequest](#)

[TradeApi.HistoryPriceRangeHistoricalRequest](#)

[TradeApi.HistoryProfileHistoricalRequest](#)

[TradeApi.HistoryRenkoHistoricalRequest](#)

[TradeApi.HistoryTickHistoricalRequest](#)

[TradeApi.HistoryTimeHistoricalRequest](#)
[TradeApi.HistoryVolumeHistoricalRequest](#)

HistoricalRequest Constructor

Historical Request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public HistoricalRequest(Copy
    Instrument instrument,
    DataType dataType
)
```

Parameters

instrument [Instrument](#)

Instrument based on which request will
be processed

dataType [DataType](#)

Data type based on which request will
be processed

▪ Example

C#

```
using Runtime.Script; Copy
using TradeApi.Trading;
using TradeApi;
using TradeApi.History;
```

```
namespace TestEnv
{
    public class HistoricalRequestExample : StrategyBuilder
    {
        public HistoricalRequestExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "HistoricalRequestExample";
            #endregion
        }

        public override void Init()
        {
            var newHistoricalReq = new
                TimeHistoricalRequest(instrument:
                    InstrumentsManager.Current,
                    dataType: DataType.Trade,
                    period: Period.Hour, value: 15
                );
            var interval =
                new
                Interval(DateTime.UtcNow.AddDays(-2), DateTime.UtcNow);

            HistoricalDataManager.OnLoaded
                +=
                this.HistoricalDataManager_OnLoaded;
        }
    }
}
```

```
    HistoricalDataManager.Get(newH
    istoricalReq, interval);
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var
low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}
```

```
        public override  
void Update(TickStatus args)  
    {  
        //  
    }  
}
```

◀ See Also

[Reference](#)

[HistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HistoricalRequest Properties

The [HistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 DataType	Represent the history data type
 Instrument	Represents the Instrument for selected historical data
 LoadExtendedSession	Load Extended Session history
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes)

[Top](#)

See Also

Reference

[HistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HistoricalRequestData Type Property

Represent the history data type

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DataType DataType {  
    get; }
```

[Copy](#)

Property Value

[DataType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
        DataTypeExample :  
            StrategyBuilder  
    {  
        public  
            DataTypeExample()  
    }
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "DataTypeExample";
        #endregion
    }

        public override
void Init()
{
    if
(HistoryDataSeries.HistoricalR
equest.DataType ==
DataType.Trade)
{
    InstrumentsManager.OnTrades +=

(trade) => {

    Notification.Comment($"Instrument's
{trade.Instrument.Symbol} is
bought");
}
}

        public override
void Update(TickStatus args)
{
}

}

}
```

See Also

Reference

[HistoricalRequest Class](#)

[TradeApi.History Namespace](#)

Historical RequestInstrument Property

Represents the Instrument for selected historical data

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Instrument { Copy
    get; }
```

Property Value
[Instrument](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi;

namespace TestEnv
{
    public class
InstrumentExample :  
StrategyBuilder
```

```
    {
        public
InstrumentExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"InstrumentExample";
#endregion
}

public override
void Init()
{
    if
(HistoryDataSeries.HistoricalR
equest.Instrument.TradingStatu
s ==
TradeApi.Instruments.TradingSt
atus.Open)
{
    Notification.Comment($"
{HistoryDataSeries.HistoricalR
equest.Instrument.Symbol} is
ready to be traded");
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[HistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HistoricalRequestLoad ExtendedSession Property

Load Extended Session history

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool  
LoadExtendedSession { get;  
set; }
```

[Copy](#)

Property Value

Boolean

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System;  
  
namespace TestEnv  
{  
    public class  
LoadExtendedSessionExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
LoadExtendedSessionExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"LoadExtendedSessionExample";
        #endregion
    }

        public override
void Init()
{
    var
newHistoricalReq = new
TimeHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
period: Period.Hour, value: 15
);

newHistoricalReq.LoadExtendedS
ession = false;

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
    var data =
HistoricalDataManager.Get(newH
istoricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}
}
```

```
        private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var
low =
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}

public override
void Update(TickStatus args)
```

```
    {  
        }  
    }  
}
```

◀ See Also

Reference

[HistoricalRequest Class](#)

[TradeApi.History Namespace](#)

Historical RequestSubscribeTo Quotes Property

Subscribe to quotes (history will be change on quotes)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool SubscribeToQuotes  
{ get; set; }
```

[Copy](#)

Property Value
Boolean

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System;  
  
namespace TestEnv  
{  
    public class  
SubscribeToQuotesExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
SubscribeToQuotesExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SubscribeToQuotesExample";
        #endregion
    }

        public override
void Init()
{
    var
newHistoricalReq = new
TimeHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
period: Period.Hour, value: 15
);

newHistoricalReq.SubscribeQu
otes = false;

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
    var data =
HistoricalDataManager.Get(newH
istoricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}
}
```

```
        private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var
low =
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}

public override
void Update(TickStatus args)
```

```
    {  
    }  
}  
}
```

◀ See Also

Reference

[HistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HistoricalRequest Methods

The [HistoricalRequest](#) type exposes the following members.

▲ Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)

 GetHashCode	Serves as the default hash function. (Inherited from Object)
  GetRequest	Returns filled historical request container with given instrument name
 GetType	Gets the Type of the current instance. (Inherited from Object)
 MemberwiseClone	Creates a shallow copy of the current Object . (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[HistoricalRequest Class](#)

[TradeApi.History Namespace](#)

HistoricalRequestGet Request Method

Returns filled historical request container with given instrument name

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public HistoricalRequest  
GetRequest(  
    Instrument instrument  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

Return Value

[HistoricalRequest](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi;  
using TradeApi.History;  
using System;
```

[Copy](#)

```
using System.Linq;

namespace TestEnv
{
    public class
HistoricalRequestExample :  
StrategyBuilder
    {
        public
HistoricalRequestExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"HistoricalRequestExample";
            #endregion
        }

        public override
void Init()
        {
            var
extrathistoricalReq =
HistoryDataSeries.HistoricalRe
quest.GetRequest(InstrumentsMa
nager.Current);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
            var data =
HistoricalDataManager.Get(extr
athistoricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
        }
    }
}
```

```
        private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var
low =
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}

public override
void Update(TickStatus args)
```

```
    {  
        }  
    }  
}
```

◀ See Also

Reference

[HistoricalRequest Class](#)

[TradeApi.History Namespace](#)

Interval Class

Represents Datetime limits for historical data queries

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryInterval](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class Interval
```

[Copy](#)

The [Interval](#) type exposes the following members.

► Constructors

	Name	Description
	Interval	UTC time interval entity

[Top](#)

► Properties

	Name	Description
	FromUtc	Lower UTC time

boundary of the interval



[ToUtc](#)

Upper UTC time boundary of the interval

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

Reference

[TradeApi.History Namespace](#)

Interval Constructor

UTC time interval entity

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public Interval(  
    DateTime fromUtc,  
    DateTime toUtc  
)
```

[Copy](#)

Parameters

fromUtc [DateTime](#)

Lower UTC time boundary of the
interval

toUtc [DateTime](#)

Upper UTC time boundary of the
interval

◀ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System;
```

[Copy](#)

```
namespace TestEnv
```

```
    {
        public class
IntervalExample :
StrategyBuilder
{
    public
IntervalExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"IntervalExample";
    #endregion
}
    HistoricalData
data;

        public override
void Init()
{
    var newData
= new
TimeHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Ask,
period: Period.Minute, value:
15);
    var from =
DateTime.UtcNow.AddDays(-1);
    var to =
DateTime.UtcNow;
    var
interval = new
Interval(fromUtc: from, toUtc:
to);
    data =
HistoricalDataManager.Get(newD
```

```
    ata, interval);
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

    }
}
}
```

See Also

[Reference](#)

[Interval Class](#)

[TradeApi.History Namespace](#)

Interval Properties

The [Interval](#) type exposes the following members.

► Properties

	Name	Description
 	FromUtc	Lower UTC time boundary of the interval
 	ToUtc	Upper UTC time boundary of the interval

[Top](#)

► See Also

[Reference](#)

[Interval Class](#)

[TradeApi.History Namespace](#)

IntervalFromUtc Property

Lower UTC time boundary of the interval

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime FromUtc { get; } Copy
```

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Trading;
using TradeApi.History;
using System;

namespace TestEnv
{
    public class
FromUtcExample :
StrategyBuilder
{
    public
```

```
FromUtcExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"FromUtcExample";
    #endregion
}

    HistoricalData
data;
    Interval interval;

        public override
void Init()
{
    var newData
= new
TimeHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Ask,
period: Period.Minute, value:
15);
    var from =
DateTime.UtcNow.AddDays(-1);
    var to =
DateTime.UtcNow;
    interval =
new Interval(fromUtc: from,
toUtc: to);
    data =
HistoricalDataManager.Get(newD
ata, interval);
}

        public override
void Update(TickStatus args)
```

```
        {
            if (args == TickStatus.IsBar)
            {
                if (HistoryDataSeries.GetTimeUtc(
                    data.Count - 1) <
                interval.FromUtc)
                {
                    var newInterval = new Interval(fromUtc:
                        HistoryDataSeries.GetTimeUtc(d
                            ata.Count - 1), toUtc:
                            DateTime.UtcNow);

                    HistoricalDataManager.ChangeIn
                    terval(data, newInterval);
                }
            }
        }
    }
```



See Also

[Reference](#)

[Interval Class](#)

[TradeApi.History Namespace](#)

IntervalToUtc Property

Upper UTC time boundary of the interval

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public DateTime ToUtc { get; } Copy
```

Property Value

[DateTime](#)

◀ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System;

namespace TestEnv
{
    public class
ToUtcExample : StrategyBuilder
    {
        public
ToUtcExample()
            : base()
        {
            #region
```

```
Initialization

    Credentials.ProjectName =
    "ToUtcExample";
                #endregion
}

        HistoricalData
data;
        Interval interval;

    public override
void Init()
{
    var newData
= new
TimeHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Ask,
period: Period.Minute, value:
15);
    var from =
DateTime.UtcNow.AddDays(-1);
    var to =
DateTime.UtcNow;
    interval =
new Interval(fromUtc: from,
toUtc: to);
    data =
HistoricalDataManager.Get(newD
ata, interval);
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
```

```
        if
(HistoryDataSeries.GetTimeUtc(
0) > interval.ToUtc)
{
    var
newInterval = new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-1),
toUtc: DateTime.UtcNow);

HistoricalDataManager.ChangeIn
terval(data, newInterval);
}
}
}
}
```

See Also

[Reference](#)

[Interval Class](#)

[TradeApi.History Namespace](#)

Interval Methods

The [Interval](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[Interval Class](#)

[TradeApi.History Namespace](#)

KagiHistoricalRequest Class

Kagi based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryKagiHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
KagiHistoricalRequest :  
HistoricalRequest
```

[Copy](#)

The [KagiHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 KagiHistoricalRequest	Kagi based historical request

[Top](#)

Properties

Name	Description
 BuildFrom	Represents period type based on which kagi block will be generated
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 Reversal	The amount of price movement required to shift a chart to the right. This condition is used on charts that only take into consideration price movement instead of both price and time



SubscribeToQuotes

Subscribe to quotes (history will be change on quotes)
(Inherited from [HistoricalRequest](#))



Value

Represent the amount of historical elements

[Top](#)

Methods

	Name	Description
≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)



GetType

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.History Namespace](#)

KagiHistoricalRequest Constructor

Kagi based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public KagiHistoricalRequest(  
    Instrument instrument,  
    DataType dataType,  
    BuildFrom buildFrom,  
    int value,  
    int reversal  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

Instrument based on which request will
be processed

dataType [DataType](#)

Data type based on which request will
be processed

buildFrom [BuildFrom](#)

represents period type based on which
kagi block will be generated

value [Int32](#)

amount of historical elements based on
which kagi block will be generated

reversal Int32

The amount of price movement required to shift a chart to the right

Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
KagiHistoricalRequestExample : StrategyBuilder
    {
        public
KagiHistoricalRequestExample()
            : base()
        {
            #region
Initialization

Credentials.ProjectName =
"KagiHistoricalRequestExample"
;
            #endregion
        }

        public override
void Init()
{
    var
HistoricalReq = new
KagiHistoricalRequest(instrume
nt:
```

Copy

```
InstrumentsManager.Current,
dataType: DataType.Trade,
            buildFrom:
BuildFrom.Ticks, value: 1,
reversal: 1);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
    }

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
    {
        for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
        {
            var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
            var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
            var low
=
```

```
loadedHistoricalData.GetValue(
PriceType.Low, i);
                                var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}

public override
void Update(TickStatus args)
{
}

}

}
```

See Also

Reference

[KagiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

KagiHistoricalRequest Properties

The [KagiHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
  BuildFrom	Represents period type based on which kagi block will be generated
  DataType	Represent the history data type (Inherited from HistoricalRequest)
  Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
  LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
  Reversal	The amount of price movement required to shift a

chart to the right. This condition is used on charts that only take into consideration price movement instead of both price and time



[SubscribeToQuotes](#)

Subscribe to quotes (history will be change on quotes)
(Inherited from [HistoricalRequest](#))



[Value](#)

Represent the amount of historical elements

[Top](#)

▲ See Also

Reference

[KagiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

KagiHistoricalRequestBuildFrom Property

Represents period type based on which kagi block will be generated

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuildFrom BuildFrom {  
    get; }
```

[Copy](#)

Property Value
[BuildFrom](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
BuildFromExample :  
StrategyBuilder  
{  
    public  
BuildFromExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"BuildFromExample";  
        #endregion  
    }  
  
    public override  
void Init()  
    {  
        BuildFrom?  
period = null;  
  
if(HistoryDataSeries.Historica  
lRequest is  
TickHistoricalRequest)  
            period =  
(HistoryDataSeries.HistoricalR  
equest as  
KagiHistoricalRequest)?.BuildF  
rom;  
            var  
HistoricalReq = new  
KagiHistoricalRequest(instrume  
nt:  
InstrumentsManager.Current,  
dataType: DataType.Trade,  
            buildFrom:  
period ?? BuildFrom.Ticks,  
value: 1, reversal: 1);  
  
HistoricalDataManager.OnLoaded
```

```
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);
```

```
Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[KagiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

KagiHistoricalRequestReversalProperty

The amount of price movement required to shift a chart to the right. This condition is used on charts that only take into consideration price movement instead of both price and time

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public int Reversal { get; }
```

[Copy](#)

Property Value

[Int32](#)

▪ Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;
```

[Copy](#)

```
namespace TestEnv
```

```
    {
        public class
ReversalExample :
StrategyBuilder
{
        public
ReversalExample()
: base()
{
            #region
Initialization

Credentials.ProjectName =
"ReversalExample";
            #endregion
}

        public override
void Init()
{
            int? reversal =
null;
            if
(HistoryDataSeries.HistoricalR
equest is
KagiHistoricalRequest)
            reversal =
(HistoryDataSeries.HistoricalR
equest as
KagiHistoricalRequest)?.Revers
al;
            var
HistoricalReq = new
KagiHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
            buildFrom:
BuildFrom.Ticks, value: 1,
reversal: reversal ?? 1);
}
```

```
HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
```

```
    PriceType.Close, i);

    Notification.Print(string.Format
        ("Open: {0}, High: {1}, Low:
        {2}, Close: {3}", open, high,
        low, close));
    }
}

public override
void Update(TickStatus args)
{
    }

}
```

See Also

Reference

[KagiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

KagiHistorical RequestValue Property

Represent the amount of historical elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; }
```

[Copy](#)

Property Value
[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
{
    public
ValueExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ValueExample";
        #endregion
    }

        public override
void Init()
{
    int? value =
null;
    if
(HistoryDataSeries.HistoricalR
equest is
KagiHistoricalRequest)
        value =
(HistoryDataSeries.HistoricalR
equest as
KagiHistoricalRequest)?.Value;
    var
HistoricalReq = new
KagiHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
                buildFrom:
BuildFrom.Ticks, value: value
?? 1, reversal: 1);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
```

```
    Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow);  
}  
  
        private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
{  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
            var low  
=  
loadedHistoricalData.GetValue(  
PriceType.Low, i);  
            var  
close =  
loadedHistoricalData.GetValue(  
PriceType.Close, i);  
  
Notification.Print(string.Format  
at("Open: {0}, High: {1}, Low:  
{2}, Close: {3}", open, high,  
low, close));  
    }  
}
```

```
        }

    public override
void Update(TickStatus args)
{
    }

}
```

See Also

[Reference](#)

[KagiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

KagiHistoricalRequest Methods

The [KagiHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[KagiHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

LineBreakHistoricalRequest Class

LineBreak data based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryLineBreakHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
LineBreakHistoricalRequest :  
HistoricalRequest
```

[Copy](#)

The [LineBreakHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 LineBreakHistoricalRequest	LineBreak data based historical request

[Top](#)

Properties

Name	Description
  BuildFrom	Represents period type based on which line break will be generated
  DataType	Represent the history data type (Inherited from HistoricalRequest)
  Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
  LineBreaks	Represent the line break value
  LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
  SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
  Value	Amount of

historical
elements based
on which line
break will be
generated

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance. (Inherited from Object)



ToString

Returns a string that represents the current object.
(Inherited from Object)

[Top](#)

See Also

[Reference](#)

[TradeApi.History Namespace](#)

LineBreakHistorical Request Constructor

LineBreak data based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public  
LineBreakHistoricalRequest (  
    Instrument instrument,  
    DataType dataType,  
    BuildFrom buildFrom,  
    int value,  
    int lineBreaks  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

instrument based on which request will
be processed

dataType [DataType](#)

data type based on which request will
be processed

buildFrom [BuildFrom](#)

represents period type based on which
line break will be generated

value [Int32](#)

amount of historical elements based on
which line break will be generated

lineBreaks Int32
represent the line breaks value

Example

C#

Copy

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
LineBreakHistoricalRequestExam
ple : StrategyBuilder
    {
        public
LineBreakHistoricalRequestExam
ple()
            : base()
        {
            #region
Initialization

Credentials.ProjectName =
"LineBreakHistoricalRequestExa
mple";
            #endregion
        }

        public override
void Init()
{
    var
HistoricalReq = new
```

```
LineBreakHistoricalRequest (  
    instrument:  
    InstrumentsManager.Current,  
  
    dataType: DataType.Trade,  
  
    buildFrom: BuildFrom.Ticks,  
  
    value: 1 ,  
  
    lineBreaks: 5);  
  
HistoricalDataManager.OnLoaded  
+=  
this.HistoricalDataManager_OnL  
oaded;  
  
HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays (-2) ,  
toUtc: DateTime.UtcNow) );  
}  
  
private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue (
```

```
    PriceType.Open, i);
                var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
                var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
                var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
            }
        }
    }

public override
void Update(TickStatus args)
{
    }
}
```

See Also

Reference

[LineBreakHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

LineBreakHistoricalRequest Properties

The [LineBreakHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
  BuildFrom	Represents period type based on which line break will be generated
  DataType	Represent the history data type (Inherited from HistoricalRequest)
  Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
  LineBreaks	Represent the line break value
  LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)



SubscribeToQuotes

Subscribe to quotes (history will be change on quotes)
(Inherited from [HistoricalRequest](#))



Value

Amount of historical elements based on which line break will be generated

[Top](#)

◀ See Also

[Reference](#)

[LineBreakHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

LineBreakHistoricalRequestBuildFrom Property

Represents period type based on which line break will be generated

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuildFrom BuildFrom {  
    get; }
```

[Copy](#)

Property Value
[BuildFrom](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
BuildFromExample :  
StrategyBuilder  
{  
    public  
BuildFromExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"BuildFromExample";  
        #endregion  
    }  
  
    public override  
void Init()  
    {  
        BuildFrom?  
period = null;  
        if  
(HistoryDataSeries.HistoricalR  
equest is  
LineBreakHistoricalRequest)  
            period =  
(HistoryDataSeries.HistoricalR  
equest as  
LineBreakHistoricalRequest)?.B  
uildFrom;  
        var  
HistoricalReq = new  
LineBreakHistoricalRequest(  
  
instrument:  
InstrumentsManager.Current,  
  
dataType: DataType.Trade,  
  
buildFrom: period ??  
BuildFrom.Ticks,
```

```
        value: 1,  
  
        lineBreaks: 5);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
    }  
  
    private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
    {  
        if  
(loadedHistoricalData != null)  
        {  
            for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
            {  
                var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
                var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
                var low  
=  
loadedHistoricalData.GetValue(  
PriceType.Low, i);  
                var close =  
loadedHistoricalData.GetValue(  
PriceType.Close, i);  
                var volume =  
loadedHistoricalData.GetValue(  
PriceType.Volume, i);  
                var candle = new Candle  
                {  
                    Open = open,  
                    High = high,  
                    Low = low,  
                    Close = close,  
                    Volume = volume  
                };  
                historicalData.Add(candle);  
            }  
        }  
    }  
}
```

```
    PriceType.Low, i);  
    var  
    close =  
        loadedHistoricalData.GetValue(  
            PriceType.Close, i);  
  
    Notification.Print(string.Format  
        ("Open: {0}, High: {1}, Low:  
        {2}, Close: {3}", open, high,  
        low, close));  
    }  
}  
}  
  
public override  
void Update(TickStatus args)  
{  
}  
}  
}
```

◀ See Also

Reference

[LineBreakHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

LineBreakHistoricalRequestLineBreaks Property

Represent the line break value

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int LineBreaks { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
LineBreaksExample :
StrategyBuilder
```

```
    {
        public
LineBreaksExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"LineBreaksExample";
            #endregion
        }

        public override
void Init()
{
    int? lineBreaks
= null;
    if
(HistoryDataSeries.HistoricalR
equest is
LineBreakHistoricalRequest)
        lineBreaks
=
(HistoryDataSeries.HistoricalR
equest as
LineBreakHistoricalRequest)?.L
ineBreaks;
    var
HistoricalReq = new
LineBreakHistoricalRequest(
instrument:
InstrumentsManager.Current,

dataType: DataType.Trade,

buildFrom: BuildFrom.Ticks,

value: 1 ,
```

```
        lineBreaks: lineBreaks ?? 5);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
    }

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
    {
        var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
        var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var
low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
```

```
        close =
    loadedHistoricalData.GetValue(
    PriceType.Close, i);

    Notification.Print(string.Format
    ("Open: {0}, High: {1}, Low:
    {2}, Close: {3}", open, high,
    low, close));
    }
}

public override
void Update(TickStatus args)
{
    }

}

}
```

See Also

Reference

[LineBreakHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

LineBreakHistorical RequestValue Property

Amount of historical elements based on which line break will be generated

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; }
```

[Copy](#)

Property Value

Int32

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
{
    public
ValueExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ValueExample";
        #endregion
    }

        public override
void Init()
{
    int? value =
null;
    var
HistoricalReq = new
LineBreakHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Ticks,
value: value ?? 1 ,
lineBreaks: 5);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
    var tradeData =
HistoricalDataManager.Get(Hist
oricalReq, new
Interval(DateTime.UtcNow.AddDays(-1), DateTime.UtcNow)) as
TradeData;
```

```
        }

    private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
```

```
        public override
void Update(TickStatus args)
{
    }

}
```

See Also

Reference

[LineBreakHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

LineBreakHistoricalRequest Methods

The [LineBreakHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[LineBreakHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

Period Enumeration

Period of data

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▲ Syntax

C#

```
public enum Period
```

[Copy](#)

▲ Members

Member name	Value	Description
Second	1	Second
Minute	2	Minute
Hour	3	Hour
Day	4	Day
Week	5	Week
Month	6	Month
Year	7	Year

▲ See Also

Reference

TradeApi.History Namespace

PointAndFigureHistoricalRequest Class

PointAndFigure data based historical request

► Inheritance Hierarchy

```
SystemObject TradeApi.HistoryHistoricalRequest
  TradeApi.HistoryPointAndFigureHistoricalRequest
```

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class
PointAndFigureHistoricalRequest :
HistoricalRequest
```

[Copy](#)

The [PointAndFigureHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 PointAndFigureHistoricalRequest	PointAndFigure data based historical request

[Top](#)

► Properties

Name	Description
	

	BoxSize	Point and figure box size
	BuildFrom	Represents period type based on which point and figure block will be generated
	DataType	Represent the history data type (Inherited from HistoricalRequest)
	Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
	LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
	Reversal	The amount of price movement required to shift a chart to the right. This condition is used on charts that only take into consideration price movement instead of both price and time
	Style	Represent the history building aggregation types
	SubscribeToQuotes	Subscribe to quotes (history)

will be change on quotes)
(Inherited from [HistoricalRequest](#))



[Value](#)

Represent the amount on which point and figure will be generated

[Top](#)

◀ Methods

	Name	Description
≡◆	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡◆	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡◆ ═	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
≡◆	GetType	Gets the Type of the current instance. (Inherited from Object)
≡◆	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ **See Also**

[Reference](#)

[TradeApi.History Namespace](#)

PointAndFigure HistoricalRequest Constructor

PointAndFigure data based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public PointAndFigureHistoricalReques
t(
    Instrument instrument,
    DataType dataType,
    BuildFrom buildFrom,
    int value,
    int boxSize,
    int reversal,
    PointAndFigureStyle
    style
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

instrument based on which request will
be processed

dataType [DataType](#)

data type based on which request will be processed

buildFrom [BuildFrom](#)

represents period type based on which point and figure block will be generated

value [Int32](#)

represent the amount on which point and figure will be generated

boxSize [Int32](#)

point and figure box size

reversal [Int32](#)

the amount of price movement required to shift a chart to the right. This condition is used on charts that only take into consideration price movement instead of both price and time

style [PointAndFigureStyle](#)

represent the history building aggregation types

Example

C#

[Copy](#)

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
PointAndFigureHistoricalReques
tExample : StrategyBuilder
    {
        public
PointAndFigureHistoricalReques
tExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"PointAndFigureHistoricalReque
stExample";
        #endregion
    }

        public override
void Init()
{
    var
HistoricalReq = new
PointAndFigureHistoricalReques
t(
    instrument:
InstrumentsManager.Current,
    dataType: DataType.Trade,
    buildFrom: BuildFrom.Minute,
    value: 5,
    boxSize: 10,
    reversal: 1,
    style:
PointAndFigureStyle.Classic);

    HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
```

```
HistoricalDataManager.Get(HistoricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
```

```
        }
    }

    public override
void Update(TickStatus args)
{
    }

}

}
```

◀ ▶

▲ See Also

Reference

[PointAndFigureHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PointAndFigureHistoricalRequest Properties

The [PointAndFigureHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 BoxSize	Point and figure box size
 BuildFrom	Represents period type based on which point and figure block will be generated
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history

(Inherited from
[HistoricalRequest](#))



[Reversal](#)

The amount of price movement required to shift a chart to the right. This condition is used on charts that only take into consideration price movement instead of both price and time



[Style](#)

Represent the history building aggregation types



[SubscribeToQuotes](#)

Subscribe to quotes (history will be change on quotes)
(Inherited from [HistoricalRequest](#))



[Value](#)

Represent the amount on which point and figure will be generated

[Top](#)

See Also

[Reference](#)

[PointAndFigureHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PointAndFigure HistoricalRequestBox Size Property

Point and figure box size

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int BoxSize { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;
```

[Copy](#)

```
namespace TestEnv
```

```
{
```

```
    public class
```

```
BoxSizeExample :
StrategyBuilder
```

```
    {
        public
BoxSizeExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"BoxSizeExample";
            #endregion
        }

        public override
void Init()
        {
            int? boxSize =
null;

if(HistoryDataSeries.Historica
lRequest is
PointAndFigureHistoricalReques
t)
            boxSize =
(HistoryDataSeries.HistoricalR
equest as
PointAndFigureHistoricalReques
t)?.BoxSize;
            var
HistoricalReq = new
PointAndFigureHistoricalReques
t(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Minute,
```



```
    PriceType.High, i);
                var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
                var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}

public override
void Update(TickStatus args)
{
}

}

}
```

See Also

Reference

[PointAndFigureHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PointAndFigure HistoricalRequestBuild From Property

Represents period type based on which point and figure block will be generated

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public BuildFrom BuildFrom {  
    get; }
```

[Copy](#)

Property Value
[BuildFrom](#)

▪ Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
BuildFromExample :  
StrategyBuilder  
{  
    public  
BuildFromExample()  
        : base()  
{  
    #region  
Initialization  
  
Credentials.ProjectName =  
"BuildFromExample";  
    #endregion  
}  
  
    public override  
void Init()  
{  
    BuildFrom?  
period = null;  
  
if(HistoryDataSeries.Historica  
lRequest is  
PointAndFigureHistoricalReques  
t)  
    period =  
(HistoryDataSeries.HistoricalR  
equest as  
PointAndFigureHistoricalReques  
t)?.BuildFrom;  
    var  
HistoricalReq = new  
PointAndFigureHistoricalReques  
t(  
  
instrument:  
InstrumentsManager.Current,  
  
dataType: DataType.Trade,
```

```
        buildFrom: period ??
BuildFrom.Minute,

        value: 5,

        boxSize: 10,

        reversal: 1,

        style:
PointAndFigureStyle.Classic);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
    }

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
    {
        for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
        {
            var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
```

```
        var
    high =
loadedHistoricalData.GetValue(
PriceType.High, i);
                var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
                var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
            }
        }
    }

public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[PointAndFigureHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PointAndFigure Historical RequestReversal Property

The amount of price movement required to shift a chart to the right. This condition is used on charts that only take into consideration price movement instead of both price and time

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Reversal { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi; Copy
```

```
namespace TestEnv
{
    public class ReversalExample : StrategyBuilder
    {
        public ReversalExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "ReversalExample";
            #endregion
        }

        public override void Init()
        {
            int? reversal =
                null;

            if(HistoryDataSeries.HistoricalRequest is
                PointAndFigureHistoricalRequest)
                reversal =
                    (HistoryDataSeries.HistoricalRequest as
                        PointAndFigureHistoricalRequest)?.Reversal;
            var HistoricalReq = new
                PointAndFigureHistoricalRequest(
                    instrument:
```

```
InstrumentsManager.Current,  
    dataType: DataType.Trade,  
    buildFrom: BuildFrom.Minute,  
    value: 5,  
    boxSize: 10,  
    reversal: reversal ?? 1,  
    style:  
    PointAndFigureStyle.Classic);  
  
HistoricalDataManager.OnLoaded  
+=  
this.HistoricalDataManager_OnL  
oaded;  
  
HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
}  
  
private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var
```

```
    open =
        loadedHistoricalData.GetValue(
            PriceType.Open, i);
                var
    high =
        loadedHistoricalData.GetValue(
            PriceType.High, i);
                var low
    =
        loadedHistoricalData.GetValue(
            PriceType.Low, i);
                var
    close =
        loadedHistoricalData.GetValue(
            PriceType.Close, i);

    Notification.Print(string.Format
        ("Open: {0}, High: {1}, Low:
        {2}, Close: {3}", open, high,
        low, close));
    }
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[PointAndFigureHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PointAndFigure HistoricalRequestStyle Property

Represent the history building aggregation types

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public PointAndFigureStyle  
Style { get; }
```

[Copy](#)

Property Value
[PointAndFigureStyle](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
StyleExample : StrategyBuilder
{
    public
StyleExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"StyleExample";
        #endregion
    }

        public override
void Init()
    {

PointAndFigureStyle? style =
null;

if(HistoryDataSeries.Historica
lRequest is
PointAndFigureHistoricalReques
t)
    style =
(HistoryDataSeries.HistoricalR
equest as
PointAndFigureHistoricalReques
t)?.Style;
    var
HistoricalReq = new
PointAndFigureHistoricalReques
t(
    instrument:
InstrumentsManager.Current,
    dataType: DataType.Trade,
```

```
buildFrom: BuildFrom.Minute,  
value: 5,  
boxSize: 10,  
reversal: 1,  
style: style ??  
PointAndFigureStyle.Classic);  
  
HistoricalDataManager.OnLoaded  
+=  
this.HistoricalDataManager_OnL  
oaded;  
  
HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
}  
  
private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var
```

```
    high =
        loadedHistoricalData.GetValue(
            PriceType.High, i);
                var low
    =
        loadedHistoricalData.GetValue(
            PriceType.Low, i);
                var
close =
        loadedHistoricalData.GetValue(
            PriceType.Close, i);

Notification.Print(string.Format
    ("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[PointAndFigureHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PointAndFigure Historical RequestValue Property

Represent the amount on which point and figure will be generated

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; }
```

[Copy](#)

Property Value

Int32

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
```

[Copy](#)

```
        {
            public
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion
}

        public override
void Init()
{
    int? value =
null;

if(HistoryDataSeries.HistoricalRequest is
PointAndFigureHistoricalRequest)
    value =
(HistoryDataSeries.HistoricalRequest as
PointAndFigureHistoricalRequest)?.Value;
    var
HistoricalReq = new
PointAndFigureHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Minute,
```



```
    PriceType.High, i);
                var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
                var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[PointAndFigureHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PointAndFigure HistoricalRequest Methods

The [PointAndFigureHistoricalRequest](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of

the current instance.
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[PointAndFigureHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PointAndFigureStyle Enumeration

Type of the PnF

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum  
PointAndFigureStyle
```

[Copy](#)

► Members

Member name	Value	Description
Classic	0	Classic
HighLow	1	HighLow

► See Also

[Reference](#)

[TradeApi.History Namespace](#)

PriceRangeHistoricalRequest Class

PriceRange based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryPriceRangeHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class  
PriceRangeHistoricalRequest :  
HistoricalRequest
```

Copy

The [PriceRangeHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 PriceRangeHistoricalRequest	PriceRange data based historical request

[Top](#)

Properties

Name	Description
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 Value	Represent the value of historical elements

[Top](#)

Methods

Name	Description
 Equals	Determines

whether the specified object is equal to the current object.
(Inherited from [Object](#))



[GetHashCode](#)

Serves as the default hash function.
(Inherited from [Object](#))



[GetRequest](#)

Returns filled historical request container with given instrument name
(Inherited from [HistoricalRequest](#))



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

Reference

TradeApi.History Namespace

PriceRangeHistorical Request Constructor

PriceRange data based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public  
PriceRangeHistoricalRequest(  
    Instrument instrument,  
    DataType dataType,  
    int value  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

Instrument based on which request will
be processed

dataType [DataType](#)

Data type based on which request will
be processed

value [Int32](#)

Value of historical elements based on
which request will be processed

► Example

C#

[Copy](#)

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
PriceRangeHistoricalRequestExa
mple : StrategyBuilder
{
    public
PriceRangeHistoricalRequestExa
mple()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"PriceRangeHistoricalRequestEx
ample";
        #endregion
    }

    public override
void Init()
{
    var
HistoricalReq = new
PriceRangeHistoricalRequest(
instrument:
InstrumentsManager.Current,

dataType: DataType.Trade,

value: 1);
}
```

```
HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
```

```
    PriceType.Close, i);

    Notification.Print(string.Format
        at("Open: {0}, High: {1}, Low:
        {2}, Close: {3}", open, high,
        low, close));
    }
}

public override
void Update(TickStatus args)
{
    }

}
```

}

See Also

Reference

[PriceRangeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PriceRangeHistoricalRequest Properties

The [PriceRangeHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 Value	Represent the

value of historical elements

[Top](#)

◀ See Also

[Reference](#)

[PriceRangeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PriceRangeHistorical RequestValue Property

Represent the value of historical elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; set; } Copy
```

Property Value
[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
    {
        public
ValueExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ValueExample";
        #endregion
    }

        public override
void Init()
{
    int? value =
null;
    if
(HistoryDataSeries.HistoricalR
equest is
PriceRangeHistoricalRequest)
        value =
(HistoryDataSeries.HistoricalR
equest as
PriceRangeHistoricalRequest)?.Value;
    var
HistoricalReq = new
PriceRangeHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
value: value ?? 1);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
```

```
HistoricalDataManager.Get(HistoricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
```

```
        }
    }

    public override
void Update(TickStatus args)
{
    }

}

}
```

◀ ▶

▲ See Also

Reference

[PriceRangeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PriceRangeHistorical Request Methods

The [PriceRangeHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[PriceRangeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

PriceType Enumeration

Type of the price

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum PriceType
```

[Copy](#)

► Members

Member name	Value	Description
Open	0	Use open price.
Close	1	Use close price.
High	2	Use high price.
Low	3	Use low price.
Medium	4	Use medium price.
Typical	5	Use typical price.
Weighted	6	Use weighted price.

See Also

Reference

[TradeApi.History Namespace](#)

ProfileHistoricalRequest Class

Profile data based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryProfileHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
ProfileHistoricalRequest :  
HistoricalRequest
```

[Copy](#)

The [ProfileHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 ProfileHistoricalRequest	Profile data based historical request

[Top](#)

Properties

Name	Description
 BuildFrom	Represents the type of period
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 ProfileBasedOn	Represents period type based on which profile block will be generated
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)



ValueBuildFrom

Represent the value of historical elements



ValueProfileBasedOn

Represent the value of historical elements based on profile

[Top](#)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current

instance.
(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.History Namespace](#)

ProfileHistorical Request Constructor

Profile data based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

[Copy](#)

```
public  
ProfileHistoricalRequest(  
    Instrument instrument,  
    DataType dataType,  
    BuildFrom  
    profileBasedOn,  
    int  
    valueProfileBasedOn,  
    BuildFrom buildFrom,  
    int valueBuildFrom  
)
```

Parameters

instrument [Instrument](#)

instrument based on which request will
be processed

dataType [DataType](#)

data type based on which request will
be processed

profileBasedOn [BuildFrom](#)

represents period type based on which profile block will be generated
valueProfileBasedOn Int32
represent the value of historical elements which profile is based on
buildFrom BuildFrom
represents the type of period
valueBuildFrom Int32
represent the value of historical elements

Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ProfileHistoricalRequestExempl
e : StrategyBuilder
    {
        public
ProfileHistoricalRequestExempl
e()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"ProfileHistoricalRequestExamp
le";
            #endregion
        }
    }
}
```

```
        }

    public override
void Init()
{
    var
HistoricalReq = new
ProfileHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
profileBasedOn: BuildFrom.Day,
valueProfileBasedOn: 1,
buildFrom: BuildFrom.Week,
valueBuildFrom: 1);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
```

```
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
    {
        var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
        var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
    }
}
}

public override
void Update(TickStatus args)
{
}
}
```

See Also

[Reference](#)

[ProfileHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ProfileHistorical Request Properties

The [ProfileHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 BuildFrom	Represents the type of period
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 ProfileBasedOn	Represents period type based on which profile block will be generated

	SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
	ValueBuildFrom	Represent the value of historical elements
	ValueProfileBasedOn	Represent the value of historical elements based on profile

[Top](#)

◀ See Also

[Reference](#)

[ProfileHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ProfileHistoricalRequestBuildFromProperty

Represents the type of period

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuildFrom BuildFrom {  
    get; }
```

[Copy](#)

Property Value

[BuildFrom](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class  
BuildFromExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
BuildFromExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BuildFromExample";
    #endregion
}

        public override
void Init()
{
    BuildFrom?
period = null;
    if
(HistoryDataSeries.HistoricalR
equest is
ProfileHistoricalRequest)
        period =
(HistoryDataSeries.HistoricalR
equest as
ProfileHistoricalRequest)?.Bui
ldFrom;
    var
HistoricalReq = new
ProfileHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
profileBasedOn: BuildFrom.Day,
valueProfileBasedOn: 1,
```

```
buildFrom: period ??  
BuildFrom.Week,  
  
valueBuildFrom: 1);  
  
HistoricalDataManager.OnLoaded  
+=  
this.HistoricalDataManager_OnL  
oaded;  
  
HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
}  
  
private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
            var low  
=
```

```
loadedHistoricalData.GetValue(
PriceType.Low, i);
                                var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[ProfileHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ProfileHistoricalRequestProfileBasedOn Property

Represents period type based on which profile block will be generated

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuildFrom  
ProfileBasedOn { get; }
```

[Copy](#)

Property Value
[BuildFrom](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
ProfileBasedOnExample :  
StrategyBuilder  
{  
    public  
ProfileBasedOnExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"ProfileBasedOnExample";  
        #endregion  
    }  
  
    public override  
void Init()  
    {  
        BuildFrom?  
period = null;  
        if  
(HistoryDataSeries.HistoricalR  
equest is  
ProfileHistoricalRequest)  
            period =  
(HistoryDataSeries.HistoricalR  
equest as  
ProfileHistoricalRequest)?.Pro  
fileBasedOn;  
        var  
HistoricalReq = new  
ProfileHistoricalRequest(  
  
instrument:  
InstrumentsManager.Current,  
  
dataType: DataType.Trade,  
  
profileBasedOn: period ??  
BuildFrom.Day,
```

```
    valueProfileBasedOn: 1,  
  
    buildFrom: BuildFrom.Week,  
  
    valueBuildFrom: 1);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
}  
  
        private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
            var low
```

```
=  
loadedHistoricalData.GetValue(  
PriceType.Low, i);  
var  
close =  
loadedHistoricalData.GetValue(  
PriceType.Close, i);  
  
Notification.Print(string.Format  
at("Open: {0}, High: {1}, Low:  
{2}, Close: {3}", open, high,  
low, close));  
}  
}  
}  
  
public override  
void Update(TickStatus args)  
{  
}  
}  
}  
}
```



See Also

Reference

[ProfileHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ProfileHistorical RequestValueBuild From Property

Represent the value of historical elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int ValueBuildFrom {  
    get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
ValueBuildFromExample :
```

```
StrategyBuilder
{
    public
ValueBuildFromExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ValueBuildFromExample";
        #endregion
    }

        public override
void Init()
{
    int? value =
null;
    if
(HistoryDataSeries.HistoricalR
equest is
ProfileHistoricalRequest)
        value =
(HistoryDataSeries.HistoricalR
equest as
ProfileHistoricalRequest)?.Val
ueBuildFrom;
    var
HistoricalReq = new
ProfileHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
profileBasedOn: BuildFrom.Day,
valueProfileBasedOn: 1,
```

```
buildFrom: BuildFrom.Week,  
valueBuildFrom: value ?? 1);  
  
HistoricalDataManager.OnLoaded  
+=  
this.HistoricalDataManager_OnL  
oaded;  
  
HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
    }  
  
        private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
    {  
        if  
(loadedHistoricalData != null)  
        {  
            for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
            {  
                var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
                var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
                var low  
=  
loadedHistoricalData.GetValue(
```

```
    PriceType.Low, i);  
    var  
    close =  
        loadedHistoricalData.GetValue(  
            PriceType.Close, i);  
  
    Notification.Print(string.Format  
        ("Open: {0}, High: {1}, Low:  
        {2}, Close: {3}", open, high,  
        low, close));  
    }  
}  
}  
  
    public override  
void Update(TickStatus args)  
{  
    }  
}  
}
```

◀ See Also

Reference

[ProfileHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ProfileHistorical RequestValueProfile BasedOn Property

Represent the value of historical elements based on profile

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int ValueProfileBasedOn Copy
{ get; }
```

Property Value
Int32

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv Copy
{
    public class
```

```
ValueProfileBasedOnExample :  
StrategyBuilder  
{  
    public  
ValueProfileBasedOnExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"ValueProfileBasedOnExample";  
        #endregion  
    }  
  
    public override  
void Init()  
    {  
        int? value =  
null;  
        if  
(HistoryDataSeries.HistoricalR  
equest is  
ProfileHistoricalRequest)  
            value =  
(HistoryDataSeries.HistoricalR  
equest as  
ProfileHistoricalRequest)?.Val  
ueProfileBasedOn;  
        var  
HistoricalReq = new  
ProfileHistoricalRequest(  
  
instrument:  
InstrumentsManager.Current,  
  
dataType: DataType.Trade,  
  
profileBasedOn: BuildFrom.Day,
```

```
    valueProfileBasedOn: value ??  
    1,  
  
    buildFrom: BuildFrom.Week,  
  
    valueBuildFrom: 1);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
}  
  
        private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
            var low
```

```
=  
loadedHistoricalData.GetValue(  
PriceType.Low, i);  
var  
close =  
loadedHistoricalData.GetValue(  
PriceType.Close, i);  
  
Notification.Print(string.Format  
at("Open: {0}, High: {1}, Low:  
{2}, Close: {3}", open, high,  
low, close));  
}  
}  
}  
  
public override  
void Update(TickStatus args)  
{  
}  
}  
}  
}
```



See Also

Reference

[ProfileHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

ProfileHistorical Request Methods

The [ProfileHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[ProfileHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistoricalRequest Class

Renko data based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryRenkoHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
RenkoHistoricalRequest :  
HistoricalRequest
```

[Copy](#)

The [RenkoHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 RenkoHistoricalRequest	Renko data based historical request

[Top](#)

Properties

	Name	Description
 	BrickSize	Renko brick size
 	BuildFrom	Represents period type based on which brick will be generated
 	DataType	Represent the history data type (Inherited from HistoricalRequest)
 	Extension	Defines Extension, %
 	Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 	Inversion	Defines Inversion, %
 	LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 	NewBoxOnBreak	Defines either display new brick

on break or keep it straight

	ShowWicks	Defines either display renko brick with wicks or without
	Style	Represent the history building aggregation types
	SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
	Value	Amount of historical elements based on which brick will be generated

[Top](#)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)

	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

[TradeApi.History Namespace](#)

RenkoHistorical Request Constructor

Renko data based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public RenkoHistoricalRequest(Copy
    Instrument instrument,
    DataType dataType,
    BuildFrom buildFrom,
    int value,
    int brickSize,
    RenkoStyle style,
    bool showWicks,
    bool newBoxOnBreak,
    int extension,
    int inversion
)
```

Parameters

instrument [Instrument](#)

instrument based on which request will
be processed

dataType [DataType](#)

data type based on which request will
be processed

buildFrom [BuildFrom](#)

build from period type based on which
brick will be generated

value `Int32`
amount of historical elements based on
which brick will be generated

brickSize `Int32`
defines brick size

style `RenkoStyle`
represent the history building
aggregation types

showWicks `Boolean`
defines either display renko brick with
wicks or without

newBoxOnBreak `Boolean`
defines either display new brick on
break or keep it straight

extension `Int32`
defines Extension, %

inversion `Int32`
defines Inversion, %

Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
RenkoHistoricalRequestExample
: StrategyBuilder
{
    public
RenkoHistoricalRequestExample(
```

[Copy](#)

```
)  
    : base()  
    {  
        #region  
        Initialization  
  
        Credentials.ProjectName =  
        "RenkoHistoricalRequestExample  
        ";  
        #endregion  
    }  
  
    public override  
    void Init()  
    {  
        var  
        HistoricalReq = new  
        RenkoHistoricalRequest(  
  
            instrument:  
            InstrumentsManager.Current,  
  
            dataType: DataType.Trade,  
  
            buildFrom: BuildFrom.Ticks,  
  
            value: 1,  
  
            brickSize: 10,  
  
            style: RenkoStyle.Classic,  
  
            showWicks: false,  
  
            newBoxOnBreak: true,  
  
            extension: 12,  
  
            inversion: 100);
```

```
    HistoricalDataManager.OnLoaded
    +=
    this.HistoricalDataManager_OnL
    oaded;

    HistoricalDataManager.Get (Hist
    oricalReq, new
    Interval (fromUtc:
    DateTime.UtcNow.AddDays (-2) ,
    toUtc: DateTime.UtcNow) );
    }

        private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);
```

```
Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistorical Request Properties

The [RenkoHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 BrickSize	Renko brick size
 BuildFrom	Represents period type based on which brick will be generated
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Extension	Defines Extension, %
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 Inversion	Defines Inversion, %

 	LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 	NewBoxOnBreak	Defines either display new brick on break or keep it straight
 	ShowWicks	Defines either display renko brick with wicks or without
 	Style	Represent the history building aggregation types
 	SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 	Value	Amount of historical elements based on which brick will be generated

[Top](#)

See Also

[Reference](#)

[RenkoHistoricalRequest Class](#)

TradeApi.History Namespace

RenkoHistorical RequestBrickSize Property

Renko brick size

Namespace: [TradeApi.History](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public int BrickSize { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
BrickSizeExample :  
StrategyBuilder
```

```
        {
            public
BrickSizeExample()
 : base()
{
    #region
Initialization

Credentials.ProjectName =
"BrickSizeExample";
    #endregion
}

        public override
void Init()
{
    int? brickSize
= null;
    if
(HistoryDataSeries.HistoricalR
equest is
RenkoHistoricalRequest)
        brickSize =
(HistoryDataSeries.HistoricalR
equest as
RenkoHistoricalRequest)?.Brick
Size;
    var
HistoricalReq = new
RenkoHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Ticks,
value: 1,
```

```
        brickSize: brickSize ?? 10,  
        style: RenkoStyle.Classic,  
        showWicks: false,  
        newBoxOnBreak: true,  
        extension: 12,  
        inversion: 100);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
    }  
  
    private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
    {  
        if  
(loadedHistoricalData != null)  
        {  
            for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
            {  
                var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
        }  
    }  
}
```

```
        var
    high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
    }
}
}

public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistorical RequestBuildFrom Property

Represents period type based on which brick will be generated

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuildFrom BuildFrom {  
    get; }
```

[Copy](#)

Property Value
[BuildFrom](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
BuildFromExample :  
StrategyBuilder  
{  
    public  
BuildFromExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"BuildFromExample";  
        #endregion  
    }  
  
    public override  
void Init()  
    {  
        BuildFrom?  
period = null;  
        if  
(HistoryDataSeries.HistoricalR  
equest is  
RenkoHistoricalRequest)  
            period =  
(HistoryDataSeries.HistoricalR  
equest as  
RenkoHistoricalRequest)?.Build  
From;  
        var  
HistoricalReq = new  
RenkoHistoricalRequest(  
  
instrument:  
InstrumentsManager.Current,  
  
dataType: DataType.Trade,  
  
buildFrom: period ??  
BuildFrom.Ticks,
```

```
        value: 1,  
  
        brickSize: 20,  
  
        style: RenkoStyle.Classic,  
  
        showWicks: false,  
  
        newBoxOnBreak: true,  
  
        extension: 12,  
  
        inversion: 100);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
}  
  
        private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
{  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
{  
            var
```

```
    open =
        loadedHistoricalData.GetValue(
            PriceType.Open, i);
                var
    high =
        loadedHistoricalData.GetValue(
            PriceType.High, i);
                var low
    =
        loadedHistoricalData.GetValue(
            PriceType.Low, i);
                var
    close =
        loadedHistoricalData.GetValue(
            PriceType.Close, i);

    Notification.Print(string.Format
        ("Open: {0}, High: {1}, Low:
        {2}, Close: {3}", open, high,
        low, close));
    }
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistoricalRequestExtensionProperty

Defines Extension, %

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Extension { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ExtensionExample : StrategyBuilder
```

```
        {
            public
ExtensionExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"ExtensionExample";
    #endregion
}

        public override
void Init()
{
    int? extension
= null;
    if
(HistoryDataSeries.HistoricalR
equest is
RenkoHistoricalRequest)
        extension =
(HistoryDataSeries.HistoricalR
equest as
RenkoHistoricalRequest)?.Exten
sion;
    var
HistoricalReq = new
RenkoHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Ticks,
value: 1,
```

```
        brickSize: 10,  
  
        style: RenkoStyle.Classic,  
  
        showWicks: false,  
  
        newBoxOnBreak: true,  
  
        extension: extension ?? 12,  
  
        inversion: 100);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
    }  
  
    private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
    {  
        if  
(loadedHistoricalData != null)  
        {  
            for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
            {  
                var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
        }  
    }  
}
```

```
        var
    high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
    }
}
}

public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistorical RequestInversion Property

Defines Inversion, %

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Inversion { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
InversionExample :  
StrategyBuilder
```

```
    {
        public
InversionExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"InversionExample";
            #endregion
        }

        public override
void Init()
        {
            int? inversion
= null;
            if
(HistoryDataSeries.HistoricalR
equest is
RenkoHistoricalRequest)
                inversion =
(HistoryDataSeries.HistoricalR
equest as
RenkoHistoricalRequest)?.Inver
sion;
            var
HistoricalReq = new
RenkoHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Ticks,
value: 1,
```

```
        brickSize: 10,  
  
        style: RenkoStyle.Classic,  
  
        showWicks: false,  
  
        newBoxOnBreak: true,  
  
        extension: 12,  
  
        inversion: inversion ?? 100);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
    }  
  
    private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
    {  
        if  
(loadedHistoricalData != null)  
        {  
            for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
            {  
                var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
        }  
    }  
}
```

```
        var
    high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
    }
}
}

public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistorical RequestNewBoxOn Break Property

Defines either display new brick on break or keep it straight

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool NewBoxOnBreak {  
    get; }
```

[Copy](#)

Property Value
[Boolean](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
NewBoxOnBreakExample :  
StrategyBuilder  
{  
    public  
NewBoxOnBreakExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"NewBoxOnBreakExample";  
        #endregion  
    }  
  
    public override  
void Init()  
    {  
        bool?  
newBoxOnBreak = null;  
        if  
(HistoryDataSeries.HistoricalR  
equest is  
RenkoHistoricalRequest)  
  
newBoxOnBreak =  
(HistoryDataSeries.HistoricalR  
equest as  
RenkoHistoricalRequest)?.NewBo  
xOnBreak;  
        var  
HistoricalReq = new  
RenkoHistoricalRequest(  
  
instrument:  
InstrumentsManager.Current,  
  
dataType: DataType.Trade,  
  
buildFrom: BuildFrom.Ticks,
```

```
        value: 1,  
  
        brickSize: 10,  
  
        style: RenkoStyle.Classic,  
  
        showWicks: false,  
  
        newBoxOnBreak: newBoxOnBreak  
        ?? true,  
  
        extension: 12,  
  
        inversion: 100);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
    }  
  
    private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
    {  
        if  
(loadedHistoricalData != null)  
        {  
            for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
            {
```

```
        var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
        var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
    }
}

public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistorical RequestShowWicks Property

Defines either display renko brick with wicks or without

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool ShowWicks { get; } Copy
```

Property Value

Boolean

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ShowWicksExample :
```

```
StrategyBuilder
{
    public
ShowWicksExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ShowWicksExample";
        #endregion
    }

        public override
void Init()
    {
        var
HistoricalReq = new
RenkoHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Ticks,
value: 1,
brickSize: 10,
style: RenkoStyle.Classic,
showWicks: false,
newBoxOnBreak: true,
extension: 12,
```

```
    inversion: 100);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
    {
        var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
        var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
close =
```

```
loadedHistoricalData.GetValue(
    PriceType.Close, i);

Notification.Print(string.Format
    ("Open: {0}, High: {1}, Low:
    {2}, Close: {3}", open, high,
    low, close));
}

}

public override
void Update(TickStatus args)
{
}

}

}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistorical RequestStyle Property

Represent the history building aggregation types

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public RenkoStyle Style { get; } Copy
```

Property Value

[RenkoStyle](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
StyleExample : StrategyBuilder
{ Copy
```

```
        public
StyleExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"StyleExample";
    #endregion
}

        public override
void Init()
{
    RenkoStyle?
style = null;
    if
(HistoryDataSeries.HistoricalR
equest is
RenkoHistoricalRequest)
        style =
(HistoryDataSeries.HistoricalR
equest as
RenkoHistoricalRequest)?.Style
;

    var
HistoricalReq = new
RenkoHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Ticks,
value: 1,
brickSize: 10,
```

```
        style: style ??
RenkoStyle.Classic,

        showWicks: false,
        newBoxOnBreak: true,
        extension: 12,
        inversion: 100);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
    }

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
    {
        for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
        {
            var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
```

```
        var
    high =
loadedHistoricalData.GetValue(
PriceType.High, i);
        var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
        var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
    }
}
}

public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistorical RequestValue Property

Amount of historical elements based on which brick will be generated

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; }
```

[Copy](#)

Property Value

Int32

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
{
    public
```

[Copy](#)

```
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion
}

        public override
void Init()
{
    int? value =
null;
    if
(HistoryDataSeries.HistoricalR
equest is
RenkoHistoricalRequest)
        value =
(HistoryDataSeries.HistoricalR
equest as
RenkoHistoricalRequest)?.Value
;
    var
HistoricalReq = new
RenkoHistoricalRequest(
instrument:
InstrumentsManager.Current,
dataType: DataType.Trade,
buildFrom: BuildFrom.Ticks,
value: value ?? 1,
brickSize:20,
```

```
        style: RenkoStyle.Classic,  
        showWicks: false,  
        newBoxOnBreak: true,  
        extension: 12,  
        inversion: 100);  
  
    HistoricalDataManager.OnLoaded  
    +=  
    this.HistoricalDataManager_OnL  
oaded;  
  
    HistoricalDataManager.Get(Hist  
oricalReq, new  
Interval(fromUtc:  
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
}  
  
        private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var  
high =
```

```
loadedHistoricalData.GetValue(
PriceType.High, i);
                                var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
                                var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoHistorical Request Methods

The [RenkoHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[RenkoHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

RenkoStyle Enumeration

Type of the Renko

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum RenkoStyle
```

[Copy](#)

► Members

Member name	Value	Description
Classic	0	Classic
HighLow	1	HighLow
AdvancedClassic	2	AdvancedClassic
AdvancedHighLow	3	AdvancedHighLow

► See Also

[Reference](#)

[TradeApi.History Namespace](#)

TickHistoricalRequest Class

Tick based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryTickHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
    TickHistoricalRequest :  
        HistoricalRequest
```

[Copy](#)

The [TickHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 TickHistoricalRequest	Tick based historical request

[Top](#)

◀ Properties

Name	Description
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 Value	Represent the value of historical elements

[Top](#)

◀ Methods

Name	Description
	

	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

Reference

[TradeApi.History Namespace](#)

TickHistoricalRequest Constructor

Tick based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TickHistoricalRequest(  
    Instrument instrument,  
    DataType dataType,  
    int value  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

Instrument based on which request will
be processed

dataType [DataType](#)

Data type based on which request will
be processed

value [Int32](#)

Tick's value based on which request will
be processed

► Example

C#

[Copy](#)

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
TickHistoricalRequestExample :  
StrategyBuilder
    {
        public
TickHistoricalRequestExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"TickHistoricalRequestExample"
;
            #endregion
        }

        public override
void Init()
        {
            var
newHistoricalReq = new
TickHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
value: 15 );

HistoricalDataManager.OnLoaded
+ =
this.HistoricalDataManager_OnL
oaded;
```

```
HistoricalDataManager.Get(newH
istoricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
at("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
```

```
    low, close));
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[TickHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TickHistoricalRequest Properties

The [TickHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 Value	Represent the

value of historical elements

[Top](#)

See Also

[Reference](#)

[TickHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TickHistorical RequestValue Property

Represent the value of historical elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; }
```

[Copy](#)

Property Value
[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
    {
        public
ValueExample()
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"ValueExample";
            #endregion
        }

    public override
void Init()
{
    int? period =
null;
    period =
(HistoryDataSeries.HistoricalR
equest as
TickHistoricalRequest)?.Value;
    var
HistoricalReq = new
TickHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
period ?? 15);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
```

```
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var
low =
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
low, close));
}
}
}

public override
void Update(TickStatus args)
{
}
```

```
        }  
    }
```



▲ See Also

Reference

[TickHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TickHistoricalRequest Methods

The [TickHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TickHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TimeHistoricalRequest Class

Time based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryTimeHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
TimeHistoricalRequest :  
HistoricalRequest
```

[Copy](#)

The [TimeHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 TimeHistoricalRequest	Time based historical request

[Top](#)

Properties

Name	Description
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 Period	Represent the period of historical elements
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 Value	Represent the value of historical elements

[Top](#)

◀ Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
  GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.History Namespace](#)

TimeHistoricalRequest Constructor

Time based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TimeHistoricalRequest(  
    Instrument instrument,  
    DataType dataType,  
    Period period,  
    int value  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

Instrument based on which request will
be processed

dataType [DataType](#)

Data type based on which request will
be processed

period [Period](#)

Period type based on which request will
be processed

value [Int32](#)

Period's value based on which request
will be processed

Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
TimeHistoricalRequestExample : StrategyBuilder
    {
        public
TimeHistoricalRequestExample()
            : base()
        {
            #region
Initialization
            Credentials.ProjectName =
"TimeHistoricalRequestExample";
        ;
            #endregion
        }

        public override
void Init()
        {
            var
newHistoricalReq = new
TimeHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
period: Period.Hour, value: 15
);
    }
    
```

Copy

```
HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(newH
istoricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
```

```
    PriceType.Close, i);

    Notification.Print(string.Format
        ("Open: {0}, High: {1}, Low:
        {2}, Close: {3}", open, high,
        low, close));
    }
}

public override
void Update(TickStatus args)
{
    }

}
```

See Also

Reference

[TimeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TimeHistoricalRequest Properties

The [TimeHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
  DataType	Represent the history data type (Inherited from HistoricalRequest)
  Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
  LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
  Period	Represent the period of historical elements
  SubscribeToQuotes	Subscribe to quotes (history will be change on

quotes)
(Inherited from
[HistoricalRequest](#))



Value

Represent the
value of historical
elements

[Top](#)

See Also

[Reference](#)

[TimeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TimeHistoricalRequestPeriod Property

Represent the period of historical elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Period Period { get; } Copy
```

Property Value

[Period](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
PeriodExample :
StrategyBuilder
```

```
        {
            public
PeriodExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"PeriodExample";
    #endregion
}

        public override
void Init()
{
    Period? period
= null;
    if
(HistoryDataSeries.HistoricalR
equest is
TimeHistoricalRequest)
        period =
(HistoryDataSeries.HistoricalR
equest as
TimeHistoricalRequest)?.Period
;
    var
HistoricalReq = new
TimeHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
period: period ?? Period.Hour,
value: 15);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
```

```
HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
```

```
    low, close));
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[TimeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TimeHistorical RequestValue Property

Represent the value of historical elements

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Value { get; }
```

[Copy](#)

Property Value
[Int32](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
ValueExample : StrategyBuilder
{
    public
ValueExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ValueExample";
        #endregion
    }

        public override
void Init()
{
    int? value =
null;
    if
(HistoryDataSeries.HistoricalR
equest is
TimeHistoricalRequest)
        value =
(HistoryDataSeries.HistoricalR
equest as
TimeHistoricalRequest)?.Value;
    var
HistoricalReq = new
TimeHistoricalRequest(instrume
nt:
InstrumentsManager.Current,
dataType: DataType.Trade,
period: Period.Hour, value:
value ?? 1);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
```

```
DateTime.UtcNow.AddDays(-2),  
toUtc: DateTime.UtcNow));  
}  
  
private void  
HistoricalDataManager_OnLoaded  
(HistoricalData  
loadedHistoricalData)  
{  
    if  
(loadedHistoricalData != null)  
    {  
        for (int i  
= loadedHistoricalData.Count -  
1; i >= 0; i--)  
        {  
            var  
open =  
loadedHistoricalData.GetValue(  
PriceType.Open, i);  
            var  
high =  
loadedHistoricalData.GetValue(  
PriceType.High, i);  
            var low  
=  
loadedHistoricalData.GetValue(  
PriceType.Low, i);  
            var  
close =  
loadedHistoricalData.GetValue(  
PriceType.Close, i);  
  
Notification.Print(string.Format  
at("Open: {0}, High: {1}, Low:  
{2}, Close: {3}", open, high,  
low, close));  
    }  
}
```

```
        public override  
void Update(TickStatus args)  
    {  
        //  
    }  
}
```

See Also

[Reference](#)

[TimeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TimeHistoricalRequest Methods

The [TimeHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TimeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TradeData Class

Trade data entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalData](#)
[TradeApi.HistoryTradeData](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class TradeData :  
    HistoricalData
```

[Copy](#)

The [TradeData](#) type exposes the following members.

► Properties

	Name	Description
	Count	The count of the historical data elements (Inherited from HistoricalData)
	HistoricalRequest	Reference to the initial HistoricalRequest of current HistoricalData (Inherited from HistoricalData)

[Top](#)

Methods

	Name	Description
≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡ ⚡	FindInterval	Index of the historical unit by given time in UTC (Inherited from HistoricalData)
≡ ⚡	GetAggressor	Returns the trade operation side name with given offset bar number
≡ ⚡	GetBuyer	Buyer's name
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡ ⚡	GetHighest	A collection of the unit indexes which are the highest values within interval (Inherited from HistoricalData)
≡ ⚡	GetLowest	A collection of the unit indexes which are the lowest values within interval (Inherited from HistoricalData)
≡ ⚡	GetPrice	Price of the historian trade
≡ ⚡	GetSeller	Seller's name
≡ ⚡	GetSize	The trade volume inside required Historical tick
≡ ⚡	GetTimeUtc	Time in UTC of historical trade (Overrides HistoricalDataGetTimeUtc(Int32))

	GetType	Gets the Type of the current instance. (Inherited from Object)
 	GetValue	Retrieves historical value by index based on price type (Inherited from HistoricalData)
 	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.History Namespace](#)

TradeData Properties

The [TradeData](#) type exposes the following members.

► Properties

Name	Description
 Count	The count of the historical data elements (Inherited from HistoricalData)
 HistoricalRequest	Reference to the initial HistoricalRequest of current HistoricalData (Inherited from HistoricalData)

[Top](#)

► See Also

[Reference](#)

[TradeData Class](#)

[TradeApi.History Namespace](#)

TradeData Methods

The [TradeData](#) type exposes the following members.

▪ Methods

	Name	Description
 	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 	FindInterval	Index of the historical unit by given time in UTC (Inherited from HistoricalData)
 	GetAggressor	Returns the trade operation side name with given offset bar number
 	GetBuyer	Buyer's name
 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetHighest	A collection of the unit indexes which are the highest values within interval (Inherited from HistoricalData)
 	GetLowest	A collection of the unit indexes which are the lowest values within interval (Inherited from HistoricalData)
 	GetPrice	Price of the historian trade
 	GetSeller	Seller's name

 	GetSize	The trade volume inside required Historical tick
 	GetTimeUtc	Time in UTC of historical trade (Overrides HistoricalDataGetTimeUtc(Int32))
	GetType	Gets the Type of the current instance. (Inherited from Object)
 	GetValue	Retrieves historical value by index based on price type (Inherited from HistoricalData)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

[TradeData Class](#)

[TradeApi.History Namespace](#)

TradeDataGet Aggressor Method

Returns the trade operation side name with given offset bar number

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public AggressorFlag
GetAggressor(
    int offset = 0
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
trade's index offset

Return Value
[AggressorFlag](#)

▪ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv
{
    public class GetAggressorExample : StrategyBuilder
    {
        public GetAggressorExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "GetAggressorExample";
            #endregion
        }
        public TradeData tradeData;
    }

    public override void Init()
    {
        this.tradeData
        = HistoryDataSeries as TradeData;
    }

    public override void Update(TickStatus args)
    {
        if (args ==
            TickStatus.IsBar &&
            (this.HistoryDataSeries is TradeData) &&
            this.tradeData.Count > 1)
        {
            var aggressor =
                tradeData.GetAggressor(1);
        }
    }
}
```

```
Notification.Print($"Last  
trade's aggressor is  
{aggressor});  
        }  
    }  
}
```

◀ See Also

Reference

[TradeData Class](#)

[TradeApi.History Namespace](#)

TradeDataGetBuyer Method

Buyer's name

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string GetBuyer(Copy
    int offset = 0
)
```

Parameters

offset [Int32](#) (Optional)
trade's index offset

Return Value

[String](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
```

```
    public class
GetBuyerExample :
StrategyBuilder
{
    public
GetBuyerExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetBuyerExample";
    #endregion
}
    public TradeData
tradeData;

    public override
void Init()
{
    this.tradeData
= HistoryDataSeries as
TradeData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
TradeData) &&
this.tradeData.Count > 1)
{
    var buyer =
tradeData.GetBuyer(1);

Notification.Print($"Last
trade's buyer is {buyer}");
}
```

```
        }  
    }  
}
```

See Also

Reference

[TradeData Class](#)

[TradeApi.History Namespace](#)

TradeDataGetPrice Method

Price of the historian trade

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetPrice(  
    int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
trade's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
GetPriceExample : 
StrategyBuilder
{
    public
GetPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetPriceExample";
    #endregion
}
    public TradeData
tradeData;

    public override
void Init()
{
    this.tradeData
= HistoryDataSeries as
TradeData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
TradeData) &&
this.tradeData.Count > 1)
{
    var price =
tradeData.GetPrice(1);

Notification.Print($"Last
trade's price is {price}");
}
```

```
        }  
    }  
}
```

See Also

Reference

[TradeData Class](#)

[TradeApi.History Namespace](#)

TradeDataGetSeller Method

Seller's name

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string GetSeller(Copy
                        int offset = 0
                    )
```

Parameters

offset [Int32](#) (Optional)
trade's index offset

Return Value

[String](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
```

```
    public class
GetSellerExample : 
StrategyBuilder
{
    public
GetSellerExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetSellerExample";
    #endregion
}
    public TradeData
tradeData;

    public override
void Init()
{
    this.tradeData
= HistoryDataSeries as
TradeData;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
TradeData) &&
this.tradeData.Count > 1)
{
    var seller
= tradeData.GetSeller(1);

Notification.Print($"Last
trade's seller is {seller}");
}
```

```
        }  
    }  
}
```

See Also

Reference

[TradeData Class](#)

[TradeApi.History Namespace](#)

TradeDataGetSize Method

The trade volume inside required Historical tick

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetSize(Copy
                      int offset = 0
                    )
```

Parameters

offset [Int32](#) (Optional)
trade's index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
```

```
    {
        public class
GetPriceExample :
StrategyBuilder
{
        public
GetPriceExample()
: base()
{
            #region
Initialization

Credentials.ProjectName =
"GetSizeExample";
            #endregion
}
        public TradeData
tradeData;
    }

        public override
void Init()
{
            this.tradeData
= HistoryDataSeries as
TradeData;
}

        public override
void Update(TickStatus args)
{
            if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
TradeData) &&
this.tradeData.Count > 1)
{
            var size =
tradeData.GetSize(1);

Notification.Print($"Last
```

```
    trade's size is {size}");  
}  
}  
}
```

See Also

Reference

[TradeData Class](#)

[TradeApi.History Namespace](#)

TradeDataGetTimeUtc Method

Time in UTC of historical trade

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override DateTime  
GetTimeUtc(  
            int offset = 0  
)
```

[Copy](#)

Parameters

offset [Int32](#) (Optional)
trade's index offset

Return Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv
```

[Copy](#)

```
    {
        public class
GetTimeUtcExample :
StrategyBuilder
{
    public
GetTimeUtcExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"GetTimeUtcExample";
    #endregion
}
    public TradeData
tradeData;

        public override
void Init()
{
    this.tradeData
= HistoryDataSeries as
TradeData;
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
(this.HistoryDataSeries is
TradeData) &&
this.tradeData.Count > 1)
    {
        var time =
tradeData.GetTimeUtc(1);

Notification.Print($"Last
```

```
    trade's time is {time}");  
}  
}  
}
```

▲ See Also

[Reference](#)

[TradeData Class](#)

[TradeApi.History Namespace](#)

VolumeHistoricalRequest Class

Volume data based historical request

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.HistoryHistoricalRequest](#)

[TradeApi.HistoryVolumeHistoricalRequest](#)

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
VolumeHistoricalRequest :  
HistoricalRequest
```

[Copy](#)

The [VolumeHistoricalRequest](#) type exposes the following members.

► Constructors

Name	Description
 VolumeHistoricalRequest	Volume based historical request

[Top](#)

► Properties

	Name	Description
 	DataType	Represent the history data type (Inherited from HistoricalRequest)
 	Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 	LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 	SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 	ValueInLots	Represent the value of lots to be requested

[Top](#)

► Methods

	Name	Description
--	-------------	--------------------

 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
  GetHashCode	Serves as the default hash function. (Inherited from Object)
  GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

Reference

[TradeApi.History Namespace](#)

VolumeHistorical Request Constructor

Volume based historical request

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

[Copy](#)

```
public  
VolumeHistoricalRequest(  
    Instrument instrument,  
    DataType dataType,  
    int valueInLots  
)
```

Parameters

instrument [Instrument](#)

Instrument based on which request will
be processed

dataType [DataType](#)

Data type based on which request will
be processed

valueInLots [Int32](#)

value of lots based on which request
will be processed

► Example

C#

[Copy](#)

```
using System;
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi;

namespace TestEnv
{
    public class
VolumeHistoricalRequestExample
: StrategyBuilder
{
    public
VolumeHistoricalRequestExample
()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"VolumeHistoricalRequestExampl
e";
    #endregion
}

    public override
void Init()
{
    var
HistoricalReq = new
VolumeHistoricalRequest(instru
ment:
InstrumentsManager.Current,
dataType: DataType.Trade,
valueInLots: 1);

HistoricalDataManager.OnLoaded
+=
```

```
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var
low =
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
```

```
        at("Open: {0}, High: {1}, Low: {2}, Close: {3}", open, high,
        low, close));
    }
}

public override
void Update(TickStatus args)
{
    }

}
```

}

See Also

Reference

[VolumeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

VolumeHistorical Request Properties

The [VolumeHistoricalRequest](#) type exposes the following members.

Properties

Name	Description
 DataType	Represent the history data type (Inherited from HistoricalRequest)
 Instrument	Represents the Instrument for selected historical data (Inherited from HistoricalRequest)
 LoadExtendedSession	Load Extended Session history (Inherited from HistoricalRequest)
 SubscribeToQuotes	Subscribe to quotes (history will be change on quotes) (Inherited from HistoricalRequest)
 ValueInLots	Represent the

value of lots to be requested

[Top](#)

◀ See Also

[Reference](#)

[VolumeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

VolumeHistorical RequestValueInLots Property

Represent the value of lots to be requested

Namespace: [TradeApi.History](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int ValueInLots { get;  
}
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi;  
  
namespace TestEnv  
{  
    public class  
ValueInLotsExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
ValueInLotsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueInLotsExample";
    #endregion
}

        public override
void Init()
{
    int? value =
null;
    if
(HistoryDataSeries.HistoricalR
equest is
VolumeHistoricalRequest)
        value =
(HistoryDataSeries.HistoricalR
equest as
VolumeHistoricalRequest)?.Valu
eInLots;
    var
HistoricalReq = new
VolumeHistoricalRequest(instru
ment:
InstrumentsManager.Current,
dataType: DataType.Trade,
valueInLots: value ?? 1);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;
```

```
HistoricalDataManager.Get(Hist
oricalReq, new
Interval(fromUtc:
DateTime.UtcNow.AddDays(-2),
toUtc: DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    if
(loadedHistoricalData != null)
{
    for (int i
= loadedHistoricalData.Count -
1; i >= 0; i--)
{
    var
open =
loadedHistoricalData.GetValue(
PriceType.Open, i);
    var
high =
loadedHistoricalData.GetValue(
PriceType.High, i);
    var low
=
loadedHistoricalData.GetValue(
PriceType.Low, i);
    var
close =
loadedHistoricalData.GetValue(
PriceType.Close, i);

Notification.Print(string.Format
("Open: {0}, High: {1}, Low:
{2}, Close: {3}", open, high,
```

```
    low, close));
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[VolumeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

VolumeHistorical Request Methods

The [VolumeHistoricalRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetRequest	Returns filled historical request container with given instrument name (Inherited from HistoricalRequest)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string
that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[VolumeHistoricalRequest Class](#)

[TradeApi.History Namespace](#)

TradeApi.Indicators Namespace

► Classes

Class	Description
 BarStampColor	Bar stamp color entity
 BarStampGraphicPath	Bar stamp graphic path entity
 BarStampHorizontalLine	Bar stamp horizontal line entity
 BarStampImage	Bar stamp image entity
 BarStampPrimitiveFigure	Bar stamp primitive figure entity
 BarStamps	BarStamp entity
 BarStampText	Bar stamp text entity
 BuiltInIndicator	Class which grants

access to
BuiltIn
indicators

	BuiltInIndicators	Built in indicators entity
	BuiltInLine	Built in line entity
	BuiltInLines	Built in lines entity
	Chart	Chart entity
	Cloud	Cloud entity
	Clouds	Clouds entity
	IndicatorBuilder	Provides access to the indicator's kit of the platform
	IndicatorLine	IndicatorLine entity
	IndicatorsManager	Manager dedicated to maintain all of indicators
	Lines	Lines entity
	Markers	Markers entity

	Threshold	Threshold entity
	Thresholds	Thresholds entity
	Window	Window entity

► Structures

Structure	Description
	TimeValue TimeValue entity

► Interfaces

Interface	Description
 IBarStampElement	IBarStampElement interface

► Enumerations

Enumeration	Description
 BarStampType	BarStamp types
 ChartDataType	Chart data source type
 GradientMode	Cloud's gradient mode
 LineStyle	Lines style types

	MAMode	Moving avarage mode type
	PrimitiveFigure	Primitive figure type
	RSIMode	RSI mode type
	Shift	Supported line's shift

BarStampColor Class

Bar stamp color entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsBarStampColor](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
BarStampColor :  
IBarStampElement
```

[Copy](#)

The [BarStampColor](#) type exposes the following members.

▪ Constructors

	Name	Description
	BarStampColor	Sets bar stamp over chart's bar

[Top](#)

▪ Properties

	Name	Description
	BarColor	Bar body color of the barstamp
	BarIndex	BarIndex of the barstamp
	Type	Type of the barstamp
	WicksColor	Wicks color of the barstamp

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BarStampColor Constructor

Sets bar stamp over chart's bar

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampColor(Copy
    Color barColor,
    Color wicksColor,
    int barIndex
)
```

Parameters

barColor [Color](#)

bar body color of the barstamp

wicksColor [Color](#)

wicks color of the barstamp

barIndex [Int32](#)

barIndex of the barstamp

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
```

```
using TradeApi.History;

namespace TestEnv
{
    public class BarStampColorExample : IndicatorBuilder
    {
        public BarStampColorExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "BarStampColorExample";
            #endregion
        }

        base.SeparateWindow = false;
    }

    BuiltInIndicator ema;

    public override void Init()
    {
        ema =
            IndicatorsManager.BuildIn.MA(H
                istoryDataSeries, 200,
                MAMode.EMA);
    }

    public override void Update(TickStatus args)
    {
        if
            (ema.GetValue() >
```

```
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
BarStamps.Set(new  
BarStampColor(Color.Green,  
Color.Green, 0));  
else if  
(ema.GetValue() <  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
BarStamps.Set(new  
BarStampColor(Color.Red,  
Color.Red, 0));  
}  
}  
}
```

See Also

Reference

[BarStampColor Class](#)

[TradeApi.Indicators Namespace](#)

BarStampColor Properties

The [BarStampColor](#) type exposes the following members.

Properties

	Name	Description
 	BarColor	Bar body color of the barstamp
 	BarIndex	BarIndex of the barstamp
 	Type	Type of the barstamp
 	WicksColor	Wicks color of the barstamp

[Top](#)

See Also

[Reference](#)

[BarStampColor Class](#)

[TradeApi.Indicators Namespace](#)

BarStampColorBar Color Property

Bar body color of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color BarColor { get; } Copy
```

Property Value

Color

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.History;

namespace TestEnv
{
    public class
BarColorExample :
IndicatorBuilder
{
    public
BarColorExample()
```

```
        : base()
    {
        #region Initialization

        Credentials.ProjectName =
"BarColorExample";
        #endregion

        base.SeparateWindow = false;
    }

        BuiltInIndicator
ema;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{
    if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

    BarStamps.Set(new
BarStampColor(Color.Green,
Color.Green, 0));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))
```

```
BarStamps.Set(new  
BarStampColor(Color.Red,  
Color.Red, 0));  
else  
  
BarStamps.Set(new  
BarStampColor(Color.Gray,  
Color.Gray, 0));  
}  
}  
}
```

See Also

Reference

[BarStampColor Class](#)

[TradeApi.Indicators Namespace](#)

BarStampColorBar Index Property

BarIndex of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int BarIndex { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.History;

namespace TestEnv
{
    public class
BarIndexExample :
IndicatorBuilder
{
    public
BarIndexExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "BarIndexExample";
        #endregion

        base.SeparateWindow = false;
    }

        BuiltInIndicator
ema;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{
    if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampColor(Color.Green,
Color.Green, 0));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))
```

```
BarStamps.Set(new  
BarStampColor(Color.Red,  
Color.Red, 0));  
else  
  
BarStamps.Set(new  
BarStampColor(Color.Gray,  
Color.Gray, 0));  
}  
}  
}
```

See Also

Reference

[BarStampColor Class](#)

[TradeApi.Indicators Namespace](#)

BarStampColorType Property

Type of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampType Type {  
    get; }
```

[Copy](#)

Property Value

[BarStampType](#)

Implements

[IBarStampElementType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
BarColorExample :  
IndicatorBuilder
```

[Copy](#)

```
        {
            public
BarColorExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BarColorExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampColor(Color.Gray,
Color.Gray, 0);
    if
(barStamp.Type ==
BarStampType.BarColor &&
ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))
```

```
BarStamps.Set(new
BarStampColor(Color.Green,
Color.Green, 0));
else if
(barStamp.Type ==
BarStampType.BarColor &&
ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampColor(Color.Red,
Color.Red, 0));
}
}
}
```

See Also

Reference

[BarStampColor Class](#)

[TradeApi.Indicators Namespace](#)

BarStampColorWicks Color Property

Wicks color of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color WicksColor { get; } Copy
```

Property Value

Color

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.History;

namespace TestEnv
{
    public class
WicksColorExample :
IndicatorBuilder
{
    public
```

```
WicksColorExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
" WicksColorExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{
    if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampColor(Color.Green,
Color.Green, 0));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
```

```
ceType.Close, 1))  
  
BarStamps.Set(new  
BarStampColor(Color.Red,  
Color.Red, 0));  
else  
  
BarStamps.Set(new  
BarStampColor(Color.Gray,  
Color.Gray, 0));  
}  
}  
}
```

See Also

[Reference](#)

[BarStampColor Class](#)

[TradeApi.Indicators Namespace](#)

BarStampColor

Methods

The [BarStampColor](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BarStampColor Class](#)

[TradeApi.Indicators Namespace](#)

BarStampGraphicPath Class

Bar stamp graphic path entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsBarStampGraphicPath](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
BarStampGraphicPath :  
IBarStampElement
```

[Copy](#)

The [BarStampGraphicPath](#) type exposes the following members.

► Constructors

Name	Description
 BarStampGraphicPath	Sets bar stamp over chart's bar

[Top](#)

Properties

	Name	Description
 	BarIndex	BarIndex of the barstamp
 	Color	Color of the barstamp
 	GraphicsPath	Graphics path of the barstamp
 	Type	Type of the barstamp
 	Value	Value to plot the graphic path

[Top](#)

Methods

	Name	Description
 	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 	GetHashCode	Serves as the default hash function. (Inherited from Object)



GetType

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BarStampGraphicPath Constructor

Sets bar stamp over chart's bar

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampGraphicPath(Copy
    GraphicsPath
    graphicsPath,
    Color color,
    int barIndex,
    double value
)
```

Parameters

graphicsPath [GraphicsPath](#)

graphics path of the barstamp

color [Color](#)

color of the barstamp

barIndex [Int32](#)

bar index of the barstamp

value [Double](#)

value to plot the graphic path

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;

namespace TestEnv
{
    public class
BarStampGraphicPathExample :
IndicatorBuilder
{
    public
BarStampGraphicPathExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"BarStampGraphicPathExample";
        #endregion

base.SeparateWindow = false;
    }

    BuiltInIndicator
ema;
    GraphicsPath
myPath;

    public override
void Init()
    {
        ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
```

```
MAMode.EMA) ;  
            Rectangle  
myEllipse = new Rectangle(20,  
20, 100, 50);  
            myPath = new  
GraphicsPath();  
  
myPath.AddEllipse(myEllipse);  
        }  
  
        public override  
void Update(TickStatus args)  
        {  
            if  
(ema.GetValue() >  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
BarStamps.Set(new  
BarStampGraphicPath(myPath,  
Color.DodgerBlue, 0,  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1));  
            else if  
(ema.GetValue() <  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
BarStamps.Set(new  
BarStampGraphicPath(myPath,  
Color.OrangeRed, 0,  
HistoryDataSeries.GetValue(Pri  
ceType.Open, 1));  
        }  
    }  
}
```



See Also

Reference

[BarStampGraphicPath Class](#)

[TradeApi.Indicators Namespace](#)

BarStampGraphicPath Properties

The [BarStampGraphicPath](#) type exposes the following members.

Properties

	Name	Description
	BarIndex	BarIndex of the barstamp
	Color	Color of the barstamp
	GraphicsPath	Graphics path of the barstamp
	Type	Type of the barstamp
	Value	Value to plot the graphic path

[Top](#)

See Also

[Reference](#)

[BarStampGraphicPath Class](#)

[TradeApi.Indicators Namespace](#)

BarStampGraphic PathBarIndex Property

BarIndex of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int BarIndex { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv
{
    public class
BarIndexExample :
IndicatorBuilder
{
```

```
        public  
BarIndexExample()  
        : base()  
{  
            #region  
Initialization  
  
Credentials.ProjectName =  
"BarIndexExample";  
            #endregion  
  
base.SeparateWindow = false;  
    }  
  
        BuiltInIndicator  
ema;  
        GraphicsPath  
myPath;  
  
        public override  
void Init()  
    {  
        ema =  
IndicatorsManager.BuildIn.MA(H  
istoryDataSeries, 200,  
MAMode.EMA);  
        Rectangle  
myEllipse = new Rectangle(20,  
20, 100, 50);  
        myPath = new  
GraphicsPath();  
  
myPath.AddEllipse(myEllipse);  
    }  
  
        public override  
void Update(TickStatus args)  
    {  
        if
```

```
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.DodgerBlue, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
    else

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.DodgerBlue, 1,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
    }
}
}
```



▲ See Also

Reference

[BarStampGraphicPath Class](#)

[TradeApi.Indicators Namespace](#)

BarStampGraphic PathColor Property

Color of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color Color { get; }
```

[Copy](#)

Property Value

[Color](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv
{
    public class
ColorExample :
IndicatorBuilder
{
```

```
        public
ColorExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ColorExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
        GraphicsPath
myPath;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
        Rectangle
myEllipse = new Rectangle(20,
20, 100, 50);
        myPath = new
GraphicsPath();

myPath.AddEllipse(myEllipse);
}

        public override
void Update(TickStatus args)
{
    if
```

```
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.DodgerBlue, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
else
BarStamps.Set(
new
BarStampGraphicPath(myPath,
Color.DodgerBlue, 1,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
}
}
}
```



▲ See Also

Reference

[BarStampGraphicPath Class](#)

[TradeApi.Indicators Namespace](#)

BarStampGraphic PathGraphicsPath Property

Graphics path of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public GraphicsPath  
GraphicsPath { get; }
```

[Copy](#)

Property Value

[GraphicsPath](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using  
System.Drawing.Drawing2D;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
GraphicsPathExample :  
IndicatorBuilder  
{  
    public  
GraphicsPathExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"GraphicsPathExample";  
        #endregion  
  
        base.SeparateWindow = false;  
    }  
  
    BuiltInIndicator  
ema;  
    GraphicsPath  
myPath;  
  
    public override  
void Init()  
    {  
        ema =  
IndicatorsManager.BuildIn.MA(H  
istoryDataSeries, 200,  
MAMode.EMA);  
        Rectangle  
myEllipse = new Rectangle(20,  
20, 100, 50);  
        myPath = new  
GraphicsPath();  
  
        myPath.AddEllipse(myEllipse);  
    }  
  
    public override
```

```
void Update(TickStatus args)
{
    if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.Red, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
    else {
        var
graphicsPath = new
GraphicsPath();

graphicsPath.AddRectangle(new
Rectangle(20, 20, 100, 50));

BarStamps.Set(new
BarStampGraphicPath(graphicsPa
th, Color.DodgerBlue, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
    }
}
}
```

See Also

[Reference](#)

[BarStampGraphicPath Class](#)

[TradeApi.Indicators Namespace](#)

BarStampGraphic PathType Property

Type of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampType Type {  
    get; }
```

[Copy](#)

Property Value

[BarStampType](#)

Implements

[IBarStampElementType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using  
System.Drawing.Drawing2D;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
TypeExample : IndicatorBuilder
{
    public
TypeExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"TypeExample";
        #endregion

        base.SeparateWindow = false;
    }

        BuiltInIndicator
ema;
        GraphicsPath
myPath;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
        Rectangle
myEllipse = new Rectangle(20,
20, 100, 50);
        myPath = new
GraphicsPath();

myPath.AddEllipse(myEllipse);
}

        public override
void Update(TickStatus args)
```

```
{  
    var barStamp =  
new  
    BarStampGraphicPath(myPath,  
Color.DodgerBlue, 0,  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1));  
    if  
(barStamp.Type ==  
BarStampType.GraphicPath &&  
ema.GetValue() >  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
    BarStamps.Set(barStamp);  
    else if  
(barStamp.Type ==  
BarStampType.GraphicPath &&  
ema.GetValue() <  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
    BarStamps.Set(new  
    BarStampGraphicPath(myPath,  
Color.OrangeRed, 0,  
HistoryDataSeries.GetValue(Pri  
ceType.Open, 1)));  
    }  
}  
}
```



See Also

Reference

[BarStampGraphicPath Class](#)

[TradeApi.Indicators Namespace](#)

BarStampGraphic PathValue Property

Value to plot the graphic path

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Value { get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;

namespace TestEnv
{
    public class
ValueExample :
IndicatorBuilder
{
```

[Copy](#)

```
        public
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
        GraphicsPath
myPath;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
        Rectangle
myEllipse = new Rectangle(20,
20, 100, 50);
        myPath = new
GraphicsPath();

myPath.AddEllipse(myEllipse);
}

        public override
void Update(TickStatus args)
{
    if
```

```
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.DodgerBlue, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
    else {
        double val =
HistoryDataSeries.GetValue(Pri
ceType.Close, 1) +
InstrumentsManager.Current.Min
imalTickSize * 10D;

BarStamps.Set(new
BarStampGraphicPath(myPath,
Color.DodgerBlue, 0, val));
    }
}
}
```

See Also

Reference

[BarStampGraphicPath Class](#)

TradeApi.Indicators Namespace

BarStampGraphicPath Methods

The [BarStampGraphicPath](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BarStampGraphicPath Class](#)

[TradeApi.Indicators Namespace](#)

BarStampHorizontalLine Class

Bar stamp horizontal line entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsBarStampHorizontalLine](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
BarStampHorizontalLine :  
IBarStampElement
```

[Copy](#)

The [BarStampHorizontalLine](#) type exposes the following members.

► Constructors

Name	Description
 BarStampHorizontalLine	Sets horizontal line stamps based on bar index to the end of

the chart
window
frame

[Top](#)

◀ Properties

	Name	Description
 	BarIndex	Bar index of the barstamp
 	Color	Color of the barstamp
 	Type	Type of the barstamp
 	Value	value to plot the horizontal line

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function.

(Inherited from
[Object](#))



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from
[Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BarStampHorizontalLine Constructor

Sets horizontal line stamps based on bar index to the end of the chart window frame

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public BarStampHorizontalLine(Copy
    Color color,
    int barIndex,
    double value
)
```

Parameters

color [Color](#)

color of the barstamp

barIndex [Int32](#)

bar index of the barstamp

value [Double](#)

value to plot the horizontal line

◀ Example

C#

```
using Runtime.Script;Copy
using TradeApi.Indicators;
```

```
using System.Drawing;

namespace TestEnv
{
    public class BarStampColorExample : IndicatorBuilder
    {
        public BarStampColorExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "BarStampHorizontalLine";
            #endregion
        }

        base.SeparateWindow = false;
    }

    public override void Init()
    {
    }

    public override void Update(TickStatus args)
    {
        if (HistoryDataSeries.Count > 200) {

            BarStamps.Set(new BarStampHorizontalLine(Color.DodgerBlue, 0,
                InstrumentsManager.Current.DayInfo.PrevClose));
        }
    }
}
```

```
BarStamps.Set(new
BarStampHorizontalLine(Color.O
rangeRed, 0,
InstrumentsManager.Current.Day
Info.Open));
}
}
}
}
```

See Also

[Reference](#)

[BarStampHorizontalLine Class](#)

[TradeApi.Indicators Namespace](#)

BarStampHorizontalLine Properties

The [BarStampHorizontalLine](#) type exposes the following members.

Properties

	Name	Description
 	BarIndex	Bar index of the barstamp
 	Color	Color of the barstamp
 	Type	Type of the barstamp
 	Value	value to plot the horizontal line

[Top](#)

See Also

[Reference](#)

[BarStampHorizontalLine Class](#)

[TradeApi.Indicators Namespace](#)

BarStampHorizontal LineBarIndex Property

Bar index of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int BarIndex { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
BarIndexExample :
IndicatorBuilder
{
    public
BarIndexExample()
    : base()
```

```
        {
            #region
Initialization

Credentials.ProjectName =
"BarIndexExample";
            #endregion

base.SeparateWindow = false;
        }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count >
200) {

BarStamps.Set(new
BarStampHorizontalLine(Color.D
odgerBlue, 0,
InstrumentsManager.Current.Day
Info.PrevClose));

BarStamps.Set(new
BarStampHorizontalLine(Color.O
rangeRed, 0,
InstrumentsManager.Current.Day
Info.Open));
    if
(HistoryDataSeries.Count >
300)
{
```

```
BarStamps.Set(new  
BarStampHorizontalLine(Color.O  
rangeRed, 1,  
InstrumentsManager.Current.Day  
Info.Open));  
}  
}  
}  
}
```

See Also

[Reference](#)

[BarStampHorizontalLine Class](#)

[TradeApi.Indicators Namespace](#)

BarStampHorizontal LineColor Property

Color of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color Color { get; }
```

[Copy](#)

Property Value

[Color](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
ColorExample :
IndicatorBuilder
{
    public
ColorExample()
    : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"ColorExample";
            #endregion

base.SeparateWindow = false;
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count >
200) {

BarStamps.Set(new
BarStampHorizontalLine(Color.D
odgerBlue, 0,
InstrumentsManager.Current.Day
Info.PrevClose));

BarStamps.Set(new
BarStampHorizontalLine(Color.O
rangeRed, 0,
InstrumentsManager.Current.Day
Info.Open));
    if
(HistoryDataSeries.Count >
300)
{
```

```
    BarStamps.Set(new  
    BarStampHorizontalLine(Color.G  
    ray, 0,  
    InstrumentsManager.Current.Day  
    Info.Open));  
    }  
}  
}  
}
```

See Also

Reference

[BarStampHorizontalLine Class](#)

[TradeApi.Indicators Namespace](#)

BarStampHorizontal LineType Property

Type of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampType Type {  
    get; }
```

[Copy](#)

Property Value

[BarStampType](#)

Implements

[IBarStampElementType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using  
System.Drawing.Drawing2D;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
TypeExample : IndicatorBuilder
```

```
        {
            public
TypeExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"BarStampHorizontalLine";
#endregion

base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    var barStamp =
new
BarStampHorizontalLine(Color.O
rangeRed, 0,
InstrumentsManager.Current.Day
Info.Open);
    if (barStamp.Type
== BarStampType.HorizontalLine
&& HistoryDataSeries.Count >
200) {

BarStamps.Set(new
BarStampHorizontalLine(Color.D
odgerBlue, 0,
InstrumentsManager.Current.Day
Info.PrevClose));
}
```

```
        BarStamps.Set(barStamp);
    }
}
}
```

See Also

[Reference](#)

[BarStampHorizontalLine Class](#)

[TradeApi.Indicators Namespace](#)

BarStampHorizontal LineValue Property

value to plot the horizontal line

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Value { get; } Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.History;

namespace TestEnv
{
    public class
ValueExample :
IndicatorBuilder
{
    public
ValueExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ValueExample";
        #endregion

        base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count >
200) {

    BarStamps.Set(new
BarStampHorizontalLine(Color.D
odgerBlue, 0,
InstrumentsManager.Current.Day
Info.PrevClose));

    BarStamps.Set(new
BarStampHorizontalLine(Color.O
rangeRed, 0,
InstrumentsManager.Current.Day
Info.Open));
    if
(HistoryDataSeries.Count >
300)
{
```

```
        double  
        val =  
        HistoryDataSeries.GetValue(Pri  
ceType.Open, 1) +  
        InstrumentsManager.Current.Min  
imalTickSize * 10D;  
  
        BarStamps.Set(new  
        BarStampHorizontalLine(Color.O  
rangeRed, 0, val));  
    }  
}  
}  
}
```

See Also

[Reference](#)

[BarStampHorizontalLine Class](#)

[TradeApi.Indicators Namespace](#)

BarStampHorizontalLine Methods

The [BarStampHorizontalLine](#) type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BarStampHorizontalLine Class](#)

[TradeApi.Indicators Namespace](#)

BarStampImage Class

Bar stamp image entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.Indicators](#)
[BarStampImage](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
BarStampImage :  
IBarStampElement
```

[Copy](#)

The [BarStampImage](#) type exposes the following members.

▪ Constructors

	Name	Description
	BarStampImage	Sets bar stamp as image over chart's bar index and value

[Top](#)

◀ Properties

	Name	Description
 	BarIndex	Bar index of the barstamp
 	Image	Image of the bar stamp
 	Type	Type of the barstamp
 	Value	Value to plot the image

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BarStampImage Constructor

Sets bar stamp as image over chart's bar index and value

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public BarStampImage(  
    Image image,  
    int barIndex,  
    double value  
)
```

[Copy](#)

Parameters

image [Image](#)

 image of the bar stamp

barIndex [Int32](#)

 bar index of the barstamp

value [Double](#)

 value to plot the image

◀ Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;
```

[Copy](#)

```
using System.Drawing;
using TradeApi.Instruments;
using TradeApi.History;

namespace TestEnv
{
    public class
BarStampImageExample : 
IndicatorBuilder
{
    public
BarStampImageExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"BarStampImageExample";
        #endregion

base.SeparateWindow = false;
    }

    Instrument
instrument;
    BuiltInIndicator
ema;
    Image buyImage;
    Image sellImage;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
```

```
        MAMode.EMA) ;
                buyImage =
Image.FromFile("SampImagBuy.jp
g");
                sellImage =
Image.FromFile("SampImagSell.j
pg");
            }

        public override
void Update(TickStatus args)
{
    if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampImage(buyImage, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampImage(sellImage, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
}
}
```

See Also

[Reference](#)

[BarStampImage Class](#)

[TradeApi.Indicators Namespace](#)

BarStampImage Properties

The [BarStampImage](#) type exposes the following members.

Properties

	Name	Description
 	BarIndex	Bar index of the barstamp
 	Image	Image of the bar stamp
 	Type	Type of the barstamp
 	Value	Value to plot the image

[Top](#)

See Also

[Reference](#)

[BarStampImage Class](#)

[TradeApi.Indicators Namespace](#)

BarStampImageBar Index Property

Bar index of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int BarIndex { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.Instruments;
using TradeApi.History;

namespace TestEnv
{
    public class
BarStampImageExample :
IndicatorBuilder
{
    public
```

```
BarStampImageExample()
    : base()
{
    #region
    Initialization

    Credentials.ProjectName =
    "BarStampImageExample";
    #endregion

    base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;
Image buyImage;
Image sellImage;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    buyImage =
Image.FromFile("SampImagBuy.jp
g");
    sellImage =
Image.FromFile("SampImagSell.j
pg");
}

public override
void Update(TickStatus args)
```

```
{  
    var barStamp =  
        new BarStampImage(buyImage, 0,  
        HistoryDataSeries.GetValue(Pri  
ceType.Close, 1));  
    if  
(ema.GetValue() >  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
    BarStamps.Set(barStamp);  
    else if  
(ema.GetValue() <  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
    BarStamps.Set(new  
BarStampImage(sellImage, 0,  
HistoryDataSeries.GetValue(Pri  
ceType.Open, 1)));  
    else {  
  
    BarStamps.Set(new  
BarStampImage(buyImage, 1,  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1)));  
    }  
}  
}  
}
```

See Also

Reference

[BarStampImage Class](#)

[TradeApi.Indicators Namespace](#)

BarStamp ImageImage Property

Image of the bar stamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Image Image { get; }
```

[Copy](#)

Property Value

[Image](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.Instruments;
using TradeApi.History;

namespace TestEnv
{
    public class
ImageExample :
IndicatorBuilder
{
    public
```

[Copy](#)

```
ImageExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ImageExample";
    #endregion

base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;
Image buyImage;
Image sellImage;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    buyImage =
Image.FromFile("SampImagBuy.jp
g");
    sellImage =
Image.FromFile("SampImagSell.j
pg");
}

public override
void Update(TickStatus args)
```

```
{  
    var barStamp =  
        new BarStampImage(buyImage, 0,  
        HistoryDataSeries.GetValue(Pri  
ceType.Close, 1));  
    if  
(ema.GetValue() >  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
    BarStamps.Set(barStamp);  
    else if  
(ema.GetValue() <  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1))  
  
    BarStamps.Set(new  
BarStampImage(sellImage, 0,  
HistoryDataSeries.GetValue(Pri  
ceType.Open, 1)));  
    else {  
  
    BarStamps.Set(new  
BarStampImage(Image.FromFile(""  
SampImagEven.jpg"), 0,  
HistoryDataSeries.GetValue(Pri  
ceType.Close, 1)));  
    }  
}  
}  
}
```

See Also

Reference

[BarStampImage Class](#)

[TradeApi.Indicators Namespace](#)

BarStampImageType Property

Type of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampType Type {  
    get; }
```

[Copy](#)

Property Value

[BarStampType](#)

Implements

[IBarStampElementType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using TradeApi.History;  
using TradeApi.Instruments;
```

[Copy](#)

```
namespace TestEnv
```

```
{
```

```
    public class
```

```
        BarStampImageExample :
```

```
IndicatorBuilder
{
    public
BarStampImageExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"BarStampImageExample";
        #endregion

base.SeparateWindow = false;
    }

Instrument
instrument;
BuiltInIndicator
ema;
Image buyImage;
Image sellImage;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    buyImage =
Image.FromFile("SampImagBuy.jp
g");
    sellImage =
Image.FromFile("SampImagSell.j
pg");
}
```

```
        public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampImage(buyImage, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1));
    if
(barStamp.Type ==
BarStampType.Image &&
ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
    else if
(barStamp.Type ==
BarStampType.Image &&
ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampImage(sellImage, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
}
}
```

See Also

Reference

[BarStampImage Class](#)

[TradeApi.Indicators Namespace](#)

BarStampImageValue Property

Value to plot the image

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Value { get; } Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.Instruments;
using TradeApi.History;

namespace TestEnv
{
    public class
ValueExample :
IndicatorBuilder
{
    public
```

```
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion

base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;
Image buyImage;
Image sellImage;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    buyImage =
Image.FromFile("SampImagBuy.jp
g");
    sellImage =
Image.FromFile("SampImagSell.j
pg");
}

public override
void Update(TickStatus args)
```

```
        {
            var barStamp =
new BarStampImage(buyImage, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1));
            if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
            else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampImage(sellImage, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
            else {
                var val =
HistoryDataSeries.GetValue(Pri
ceType.Close, 1) +
instrument.MinimalTickSize *
10D;
                BarStamps.Set(
new BarStampImage(buyImage, 0,
val));
            }
        }
    }
}
```

See Also

[Reference](#)

[BarStampImage Class](#)

[TradeApi.Indicators Namespace](#)

BarStampImage Methods

The [BarStampImage](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BarStampImage Class](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitiveFigure Class

Bar stamp primitive figure entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsBarStampPrimitiveFigure](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
BarStampPrimitiveFigure :  
IBarStampElement
```

[Copy](#)

The [BarStampPrimitiveFigure](#) type exposes the following members.

► Constructors

Name	Description
 BarStampPrimitiveFigure	Sets bar stamp as primitive figure over chart's bar

index and
value

[Top](#)

◀ Properties

	Name	Description
 	BarIndex	BarIndex of the barstamp
 	Color	Color of the primitive figure
 	Figure	Primitive figure selection
 	Type	Type of the barstamp
 	Value	Value to plot the primitive figure

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash

function.
(Inherited from
[Object](#))



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from
[Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from
[Object](#))

[Top](#)

See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitive Figure Constructor

Sets bar stamp as primitive figure over chart's bar index and value

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public BarStampPrimitiveFigure(  
    Color color,  
    int barIndex,  
    double value,  
    PrimitiveFigure figure  
= PrimitiveFigure.Circle  
)
```

[Copy](#)

Parameters

color [Color](#)

color of the primitive figure

barIndex [Int32](#)

bar index of the barstamp

value [Double](#)

value to plot the primitive figure

figure [PrimitiveFigure](#) (Optional)

primitive figure selection

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.Instruments;
using TradeApi.History;

namespace TestEnv
{
    public class
BarStampPrimitiveFigureExample
: IndicatorBuilder
{
    public
BarStampPrimitiveFigureExample
()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"BarStampPrimitiveFigureExampl
e";
    #endregion

base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;

    public override
void Init()
```

Copy

```
        {
            instrument =
InstrumentsManager.Current;
            ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
        }

    public override
void Update(TickStatus args)
{
    if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
DodgerBlue, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1),
PrimitiveFigure.TriangleUp));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1),
PrimitiveFigure.TriangleDown))
;
}
}
```

See Also

[Reference](#)

[BarStampPrimitiveFigure Class](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitive Figure Properties

The [BarStampPrimitiveFigure](#) type exposes the following members.

▲ Properties

	Name	Description
 	BarIndex	BarIndex of the barstamp
 	Color	Color of the primitive figure
 	Figure	Primitive figure selection
 	Type	Type of the barstamp
 	Value	Value to plot the primitive figure

[Top](#)

▲ See Also

[Reference](#)

[BarStampPrimitiveFigure Class](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitive FigureBarIndex Property

BarIndex of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int BarIndex { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.Instruments;
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv
{
    public class
BarIndexExample :
IndicatorBuilder
```

```
        {
            public
BarIndexExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BarIndexExample";
#endregion

base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

public override
void Update(TickStatus args)
{
    var barStamp =
new
BarStampPrimitiveFigure(Color.
Gray, 0,
HistoryDataSeries.GetValue(Pri
```

```
ceType.Close, 1),
PrimitiveFigure.TriangleUp);
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
        else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1),
PrimitiveFigure.TriangleDown))
;

        else {

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
Gray, 1,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1),
PrimitiveFigure.TriangleUp));
    }
}
}
```

See Also

Reference

[BarStampPrimitiveFigure Class](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitive FigureColor Property

Color of the primitive figure

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color Color { get; }
```

[Copy](#)

Property Value

Color

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.Instruments;
using TradeApi.History;

namespace TestEnv
{
    public class
ColorExample :
IndicatorBuilder
{
    public
```

[Copy](#)

```
ColorExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ColorExample";
    #endregion

base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

public override
void Update(TickStatus args)
{
    var barStamp =
new
BarStampPrimitiveFigure(Color.
Gray, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1),
PrimitiveFigure.TriangleUp);
```

```
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1),
PrimitiveFigure.TriangleDown))
;

else {

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
Red, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1),
PrimitiveFigure.TriangleUp));
}

}
}
```

See Also

Reference

[BarStampPrimitiveFigure Class](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitiveFigure Property

Primitive figure selection

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public PrimitiveFigure Figure  
{ get; }
```

[Copy](#)

Property Value

[PrimitiveFigure](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using TradeApi.Instruments;  
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class FigureExample :  
        IndicatorBuilder  
    {
```

```
        public
FigureExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"FigureExample";
    #endregion

base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

public override
void Update(TickStatus args)
{
    var barStamp =
new
BarStampPrimitiveFigure(Color.
Gray, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1),
```

```
PrimitiveFigure.TriangleUp);
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
        else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1),
PrimitiveFigure.TriangleDown))
;

else {

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
Gray, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1),
PrimitiveFigure.Circle));
}

}
```

See Also

Reference

[BarStampPrimitiveFigure Class](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitive FigureType Property

Type of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampType Type {  
    get; }
```

[Copy](#)

Property Value

[BarStampType](#)

Implements

[IBarStampElementType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using TradeApi.Instruments;  
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv
```

```
{
```

```
    public class
```

```
        TypeExample : IndicatorBuilder
```

```
        {
            public
TypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TypeExample";
#endregion

base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

public override
void Update(TickStatus args)
{
    var barStamp =
new
BarStampPrimitiveFigure(Color.
Gray, 0,
HistoryDataSeries.GetValue(Pri
```

```
ceType.Close, 1),
PrimitiveFigure.TriangleUp);
    if
(barStamp.Type ==
BarStampType.PrimitiveFigure
&& ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
else if
(barStamp.Type ==
BarStampType.PrimitiveFigure
&& ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1),
PrimitiveFigure.TriangleDown))
;
}
}
}
```

See Also

Reference

[BarStampPrimitiveFigure Class](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitive FigureValue Property

Value to plot the primitive figure

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Value { get; } Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi.Instruments;
using TradeApi.History;

namespace TestEnv
{
    public class
ValueExample :
IndicatorBuilder
{
    public
```

```
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion

base.SeparateWindow = false;
}

Instrument
instrument;
BuiltInIndicator
ema;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

public override
void Update(TickStatus args)
{
    var barStamp =
new
BarStampPrimitiveFigure(Color.
DodgerBlue, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1),
PrimitiveFigure.TriangleUp);
```

```
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
OrangeRed, 0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1),
PrimitiveFigure.TriangleDown))
;

else {
    var val =
HistoryDataSeries.GetValue(Pri
ceType.Close, 1) +
instrument.MinimalTickSize *
10D;

BarStamps.Set(new
BarStampPrimitiveFigure(Color.
DodgerBlue, 0, val));
}
}
}
```

See Also

Reference

[BarStampPrimitiveFigure Class](#)

[TradeApi.Indicators Namespace](#)

BarStampPrimitiveFigure Methods

The [BarStampPrimitiveFigure](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BarStampPrimitiveFigure Class](#)

[TradeApi.Indicators Namespace](#)

BarStamps Class

BatStamp entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.Indicators](#)
[BarStamps](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class BarStamps
```

[Copy](#)

The [BarStamps](#) type exposes the following members.

▪ Methods

	Name	Description
	Clear	BarStamps clear
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)

	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
 	Remove	BarStamps remove
 	Set	BarStamps set
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BarStamps Methods

The [BarStamps](#) type exposes the following members.

▪ Methods

	Name	Description
 	Clear	BarStamps clear
 	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	Remove	BarStamps remove
 	Set	BarStamps set



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

See Also

[Reference](#)

[BarStamps Class](#)

[TradeApi.Indicators Namespace](#)

BarStampsClear Method

BarStamps clear

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void Clear()
```

[Copy](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
Clearxample : IndicatorBuilder
    {
        public
Clearxample()
            : base()
        {
            #region
Initialization
```

[Copy](#)

```
Credentials.ProjectName =  
"Clearxample";  
#endregion  
  
base.SeparateWindow = false;  
}  
  
BuiltInIndicator  
ma;  
  
public override  
void Init()  
{  
    ma =  
IndicatorsManager.BuildIn.MA(H  
istoryDataSeries, 12,  
MAMode.SMA);  
}  
  
public override  
void Update(TickStatus args)  
{  
    if  
(ma.Lines[0].GetValue() <  
InstrumentsManager.Current.Day  
Info.Ask)  
  
BarStamps.Set(new  
BarStampColor(barColor:  
Color.Orange, wicksColor:  
Color.Orange, barIndex: 0));  
    else  
  
BarStamps.Set(new  
BarStampColor(barColor:  
Color.Gray, wicksColor:  
Color.Gray, barIndex: 0));  
  
    if (ma.Count <
```

```
12)
{
    BarStamps.Clear();
}
}
```

See Also

[Reference](#)

[BarStamps Class](#)

[TradeApi.Indicators Namespace](#)

BarStampsRemove Method

BarStamps remove

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void Remove(Copy
                  int barIndex = 0,
                  int to = 0
                )
```

Parameters

barIndex [Int32](#) (Optional)

left end of the range to remove

to [Int32](#) (Optional)

right end of the range to remove

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
```

```
    public class
RemoveExample :
IndicatorBuilder
{
    public
RemoveExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"RemoveExample";
        #endregion

base.SeparateWindow = false;
    }

    BuiltInIndicator
ma;

    public override
void Init()
    {
        ma =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 12,
MAMode.SMA);
    }

    public override
void Update(TickStatus args)
    {
        if
(ma.Lines[0].GetValue() <
InstrumentsManager.Current.Day
Info.Ask)

BarStamps.Set(new
```

```
    BarStampColor(barColor:  
    Color.Orange, wicksColor:  
    Color.Orange, barIndex: 0));  
    else  
  
        BarStamps.Set(new  
        BarStampColor(barColor:  
        Color.Gray, wicksColor:  
        Color.Gray, barIndex: 0));  
  
        if (ma.Count <  
12)  
    {  
  
        BarStamps.Remove(ma.Count, 0);  
    }  
}  
}  
}
```

See Also

[Reference](#)

[BarStamps Class](#)

[TradeApi.Indicators Namespace](#)

BarStampsSet Method

BarStamps set

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public void Set(Copy
                 IBarStampElement
                 barStampElement
                 )
```

Parameters

barStampElement IBarStampElement
bar stamp unit

◀ Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
RemoveExample :
IndicatorBuilder
{
```

```
        public
RemoveExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"RemoveExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ma;

        public override
void Init()
{
    ma =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 12,
MAMode.SMA);
}

        public override
void Update(TickStatus args)
{
    if
(ma.Lines[0].GetValue() <
InstrumentsManager.Current.Day
Info.Ask)

BarStamps.Set(new
BarStampColor(barColor:
Color.Orange, wicksColor:
Color.Orange, barIndex: 0));
    else
```

```
        BarStamps.Set(new
    BarStampColor(barColor:
    Color.Gray, wicksColor:
    Color.Gray, barIndex: 0));
    }
}
}
```

◀ See Also

Reference

[BarStamps Class](#)

[TradeApi.Indicators Namespace](#)

BarStampText Class

Bar stamp text entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.Indicators](#)
[BarStampText](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
BarStampText :  
IBarStampElement
```

[Copy](#)

The [BarStampText](#) type exposes the following members.

▪ Constructors

	Name	Description
	BarStampText	Sets text bar stamp over chart's bar

[Top](#)

▪ Properties

	Name	Description
	BackColor	Back color of the text barstamp
	BarIndex	BarIndex of the barstamp
	Font	Fore color of the text barstamp
	ForeColor	Fore color of the text barstamp
	Text	Text of the text barstamp
	Type	Type of the text barstamp
	Value	Value to plot the text

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function.

(Inherited from
[Object](#))



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from
[Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BarStampText Constructor

Sets text bar stamp over chart's bar

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampText(  
    string text,  
    Font font,  
    Color foreColor,  
    Color backColor,  
    int barIndex,  
    double value  
)
```

[Copy](#)

Parameters

text [String](#)

text barstamp

font [Font](#)

font of the barstamp

foreColor [Color](#)

fore color of the text barstamp

backColor [Color](#)

back color of the text barstamp

barIndex [Int32](#)

bar index of the barstamp

value [Double](#)

value to plot the text

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;

namespace TestEnv
{
    public class
BarStampTextExample :
IndicatorBuilder
{
    public
BarStampTextExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"BarStampTextExample";
        #endregion

base.SeparateWindow = false;
    }

        BuiltInIndicator
ema;
        Font font;

        public override
void Init()
```

Copy

```
        {
            ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
            font = new
Font(new FontFamily("Arial"),
14, FontStyle.Bold);
        }

    public override
void Update(TickStatus args)
{
    if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

    BarStamps.Set(new
BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
    else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

    BarStamps.Set(new
BarStampText("Sell", font,
Color.OrangeRed, Color.Olive,
0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
}
}
```

See Also

[Reference](#)

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampText Properties

The [BarStampText](#) type exposes the following members.

Properties

	Name	Description
	BackColor	Back color of the text barstamp
	BarIndex	BarIndex of the barstamp
	Font	Fore color of the text barstamp
	ForeColor	Fore color of the text barstamp
	Text	Text of the text barstamp
	Type	Type of the text barstamp
	Value	Value to plot the text

[Top](#)

See Also

Reference

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampTextBack Color Property

Back color of the text barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color BackColor { get;  
}
```

[Copy](#)

Property Value

Color

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using  
System.Drawing.Drawing2D;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
BackColorExample :  
IndicatorBuilder
```

[Copy](#)

```
        {
            public
BackColorExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BackColorExample";
#endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
Font font;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    font = new
Font(new FontFamily("Arial"),
14, FontStyle.Bold);
}

        public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
```

```
ceType.Close, 1));
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
        else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampText("Sell", font,
Color.OrangeRed, Color.Olive,
0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
        else {

BarStamps.Set(new
BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Lime, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
        }
    }
}
```

See Also

Reference

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampTextBar Index Property

BarIndex of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int BarIndex { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;

namespace TestEnv
{
    public class
BarIndexExample :
IndicatorBuilder
{
```

```
        public
BarIndexExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BarIndexExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
Font font;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    font = new
Font(new FontFamily("Arial"),
14, FontStyle.Bold);
}

        public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1));
}
```

```
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampText("Sell", font,
Color.OrangeRed, Color.Olive,
0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
else {

BarStamps.Set(new
BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 1,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1));
}
}
}
```

See Also

[Reference](#)

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampTextFont Property

Fore color of the text barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Font Font { get; }
```

[Copy](#)

Property Value

[Font](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv
{
    public class
ForeColorExample :
IndicatorBuilder
{
```

```
        public
ForeColorExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ForeColorExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
Font font;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    font = new
Font(new FontFamily("Arial"),
14, FontStyle.Bold);
}

        public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1));
}
```

```
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampText("Sell", font,
Color.OrangeRed, Color.Olive,
0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
else {
    var
newArialFont = new Font(new
FontFamily("Arial"), 16,
FontStyle.Italic);

BarStamps.Set(new
BarStampText("Buy",
newArialFont,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
}
}
```

See Also

Reference

[BarStampText Class](#)

TradeApi.Indicators Namespace

BarStampTextFore Color Property

Fore color of the text barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color ForeColor { get;  
}
```

[Copy](#)

Property Value

Color

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using  
System.Drawing.Drawing2D;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
ForeColorExample :  
IndicatorBuilder
```

[Copy](#)

```
        {
            public
ForeColorExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"ForeColorExample";
#endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
Font font;

public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    font = new
Font(new FontFamily("Arial"),
14, FontStyle.Bold);
}

public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
```

```
ceType.Close, 1));
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
        else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampText("Sell", font,
Color.OrangeRed, Color.Olive,
0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
        else {

BarStamps.Set(new
BarStampText("Buy", font,
Color.Blue, Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1)));
        }
    }
}
```

See Also

Reference

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampTextText Property

Text of the text barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Text { get; }
```

[Copy](#)

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;

namespace TestEnv
{
    public class
TextExample : IndicatorBuilder
{
    public
```

```
TextExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TextExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
Font font;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    font = new
Font(new FontFamily("Arial"),
14, FontStyle.Bold);
}

        public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1));
    if
```

```
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampText("Sell", font,
Color.OrangeRed, Color.Olive,
0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
else {

BarStamps.Set(new
BarStampText("NOTRADE", font,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1));
}
}
}
```

See Also

[Reference](#)

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampTextType Property

Type of the text barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStampType Type {  
    get; }
```

[Copy](#)

Property Value

[BarStampType](#)

Implements

[IBarStampElementType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using  
System.Drawing.Drawing2D;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
TypeExample : IndicatorBuilder
{
    public
TypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TypeExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
Font font;

    public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    font = new
Font(new FontFamily("Arial"),
14, FontStyle.Bold);
}

    public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0,
```

```
HistoryDataSeries.GetValue(PriceType.Close, 1));
        if
    (barStamp.Type ==
BarStampType.Text &&
ema.GetValue() >
HistoryDataSeries.GetValue(PriceType.Close, 1))

BarStamps.Set(barStamp);
else if
    (barStamp.Type ==
BarStampType.Text &&
ema.GetValue() <
HistoryDataSeries.GetValue(PriceType.Close, 1))

BarStamps.Set(new
BarStampText("Sell", font,
Color.OrangeRed, Color.Olive,
0,
HistoryDataSeries.GetValue(PriceType.Open, 1)));
}
}
```

See Also

Reference

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampTextValue Property

Value to plot the text

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Value { get; } Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using
System.Drawing.Drawing2D;
using TradeApi.History;

namespace TestEnv
{
    public class
ValueExample :
IndicatorBuilder
{
```

```
        public
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;
Font font;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
    font = new
Font(new FontFamily("Arial"),
14, FontStyle.Bold);
}

        public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0,
HistoryDataSeries.GetValue(Pri
ceType.Close, 1));
}
```

```
        if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(barStamp);
else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampText("Sell", font,
Color.OrangeRed, Color.Olive,
0,
HistoryDataSeries.GetValue(Pri
ceType.Open, 1)));
else {
    var val =
HistoryDataSeries.GetValue(Pri
ceType.Close, 1) +
InstrumentsManager.Current.Min
imalTickSize * 10D;

BarStamps.Set(new
BarStampText("Buy", font,
Color.CornflowerBlue,
Color.Olive, 0, val));
}
}
}
```

See Also

Reference

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampText

Methods

The [BarStampText](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BarStampText Class](#)

[TradeApi.Indicators Namespace](#)

BarStampType Enumeration

BarStamp types

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum BarStampType
```

[Copy](#)

► Members

Member name	Value	Description
BarColor	0	BarColor
PrimitiveFigure	1	PrimitiveFigure
GraphicPath	2	GraphicPath
Image	3	Image
Text	4	Text
HorizontalLine	5	HorizontalLine

► See Also

[Reference](#)

TradeApi.Indicators Namespace

BuiltInIndicator Class

Class which grants access to BuiltIn indicators

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsBuiltInIndicator](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
    BuiltInIndicator
```

[Copy](#)

The [BuiltInIndicator](#) type exposes the following members.

► Properties

	Name	Description
	Count	Gets indicator's values count
	HistoryDataSeries	Represent access to instrument's historical

data based
on current
chart



Lines

Gets a
collection of
an indicator
lines



Name

Name of an
buildin
indicator

[Top](#)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)

GetValue	Returns build in indicator value by index and bar offset
--------------------------	--



ToString	Returns a string that represents the current object. (Inherited from Object)
--------------------------	--

ToString	Returns a string that represents the current object. (Inherited from Object)
--------------------------	--

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicator Properties

The [BuiltInIndicator](#) type exposes the following members.

▲ Properties

	Name	Description
 	Count	Gets indicator's values count
 	HistoryDataSeries	Represent access to instrument's historical data based on current chart
 	Lines	Gets a collection of an indicator lines
 	Name	Name of an buildin indicator

[Top](#)

▲ See Also

Reference

[BuiltInIndicator Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorCount Property

Gets indicator's values count

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
CountExample :
IndicatorBuilder
{
    public
CountExample()
    : base()
    {
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"CountExample";
        #endregion

Lines.Set("CountExample");

base.SeparateWindow = false;
}

        BuiltInIndicator
buildIn;

        public override
void Init()
{
    buildIn =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 2,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{

Notification.Print(buildIn.Cou
nt.ToString());
}
}
```

See Also

Reference

BuiltInIndicator Class
TradeApi.Indicators Namespace

BuiltInIndicatorHistory DataSeries Property

Represent access to instrument's historical data based on current chart

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public HistoricalData  
HistoryDataSeries { get; }
```

[Copy](#)

Property Value

[HistoricalData](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
HistoryDataSeriesExample :  
IndicatorBuilder  
{  
    public
```

[Copy](#)

```
HistoryDataSeriesExample()
    : base()
{
    #region
    Initialization

    Credentials.ProjectName =
    "HistoryDataSeriesExample";
    #endregion

    Lines.Set("HistoryDataSeriesEx
    ample");

    base.SeparateWindow = false;
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 0)

    HistoryDataSeries.GetValue(PriceType.Close);
}
}
```

See Also

Reference

BuiltInIndicator Class
TradeApi.Indicators Namespace

BuiltInIndicatorLines Property

Gets a collection of an indicator lines

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInLines Lines {  
    get; }
```

[Copy](#)

Property Value

[BuiltInLines](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
        LinesExample :  
            IndicatorBuilder  
    {  
        public  
        LinesExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"LinesExample";
            #endregion

Lines.Set("LinesExample");

base.SeparateWindow = false;
        }

                BuiltInIndicator
buildIn;

            public override
void Init()
{
    buildIn =
IndicatorsManager.BuildIn.MACD
(HistoryDataSeries,
fast_ema_period: 8,
slow_ema_period:12,
signal_period: 5);

Notification.Print(buildIn.Lines.Count.ToString());
}

            public override
void Update(TickStatus args)
{
}

}
}
```

See Also

[Reference](#)

[BuiltInIndicator Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorName Property

Name of an buildin indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Name { get; }
```

[Copy](#)

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
NameExample : IndicatorBuilder
    {
        public
NameExample()
            : base()
        {
            #region
```

[Copy](#)

```
Initialization

Credentials.ProjectName =
"NameExample";
#endregion

Lines.Set("NameExample");

base.SeparateWindow = false;
}

BuiltInIndicator
buildIn;

public override
void Init()
{
    buildIn =
IndicatorsManager.BuildIn.MACD
(HistoryDataSeries,
fast_ema_period: 8,
slow_ema_period:12,
signal_period: 5);

Notification.Print(buildIn.Nam
e);
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[BuiltInIndicator Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicator Methods

The [BuiltInIndicator](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	GetValue	Returns build in indicator value by index and bar offset



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BuiltInIndicator Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorGet Value Method

Returns build in indicator value by index and bar offset

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetValue(Copy
    int offset = 0,
    int lineNumber = 0
)
```

Parameters

offset [Int32](#) (Optional)

The element offset in the historical series

lineNumber [Int32](#) (Optional)

A line number of the indicator

Return Value

[Double](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
GetValueExample : IndicatorBuilder
    {
        public
GetValueExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"GetValueExample";
            #endregion

Lines.Set("GetValueExample");

            base.SeparateWindow
= false;
        }

        BuiltInIndicator
buildIn;

        public override void
Init()
        {
            buildIn =
IndicatorsManager.BuildIn.MA(Histo
ryDataSeries, 2, MAMode.EMA);
        }

        public override void
Update(TickStatus args)
        {
```

```
        if(args ==  
    TickStatus.IsQuote)  
  
        buildIn.GetValue(offset: 0,  
    lineNumber: 0);  
        else  
  
        buildIn.GetValue(offset: 1,  
    lineNumber: 0);  
    }  
}
```

See Also

Reference

[BuiltInIndicator Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicators Class

Built in indicators entity

▪ Inheritance Hierarchy

[SystemObject](#) `TradeApi.Indicators.BuiltInIndicators`

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
    BuiltInIndicators
```

[Copy](#)

The `BuiltInIndicators` type exposes the following members.

▪ Methods

Name	Description
 AC	AC indicator
 AD	AD indicator
 ADX(HistoricalData, Int32)	ADX indicator
 ADX(HistoricalData, Int32, MAMode)	ADX indicator

 	<code>ADX(HistoricalData, Int32, MAMode, PriceType)</code>	ADX indicator
 	<code>Alligator</code>	Alligator indicator
 	<code>AO</code>	AO indicator
 	<code>ATR(HistoricalData, Int32)</code>	ATR indicator
 	<code>ATR(HistoricalData, Int32, MAMode)</code>	ATR indicator
 	<code>Bands(HistoricalData, Int32)</code>	Bands indicator
 	<code>Bands(HistoricalData, Int32, MAMode)</code>	Bands indicator
 	<code>Bands(HistoricalData, Int32, MAMode, Double)</code>	Bands indicator
 	<code>Bands(HistoricalData, FuncInt32, Double, Int32, MAMode, Double)</code>	Bands indicator
 	<code>Bands(HistoricalData, Int32, MAMode, Double, PriceType)</code>	Bands indicator
 	<code>BearsPower</code>	BearsPower indicator
 	<code>BullsPower</code>	BullsPower indicator
 	<code>BWMFI</code>	BWMFI indicator

 CCI(HistoricalData, Int32)	CCI indicator
 CCI(HistoricalData, Int32, MAMode)	CCI indicator
 CCI(HistoricalData, FuncInt32, Double, Int32, MAMode)	CCI indicator
 CCI(HistoricalData, Int32, MAMode, PriceType)	CCI indicator
 Custom	Custom indicator
 DeMarker	DeMarker indicator
 Envelopes(HistoricalData, FuncInt32, Double, Int32, MAMode, Double)	Envelopes indicator
 Envelopes(HistoricalData, Int32, MAMode, Double, PriceType)	Envelopes indicator
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Force	Force indicator

 Fractals	Fractals indicator
 Gator	Gator indicator
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 Ichimoku	Ichimoku indicator
 MA(FuncInt32, Double, Int32, MAMode)	MA indicator
 MA(HistoricalData, Int32, MAMode)	MA indicator
 MA(FuncInt32, Double, Int32, MAMode, Int32)	MA indicator
 MA(HistoricalData, Int32, MAMode, Int32, PriceType)	MA indicator
 MACD(HistoricalData, Int32, Int32, Int32)	MACD indicator
 MACD(HistoricalData, Int32, Int32, Int32,	MACD indicator

PriceType)

 	MFI	MFI indicator
 	Momentum(FuncInt32, Double, Int32)	Momentum indicator
 	Momentum(HistoricalData, Int32)	Momentum indicator
 	Momentum(HistoricalData, Int32, PriceType)	Momentum indicator
 	OBV	OBV indicator
 	OsMA	OsMA indicator
 	RSI(HistoricalData, Int32)	RSI indicator
 	RSI(FuncInt32, Double, Int32, RSIMode)	RSI indicator
 	RSI(HistoricalData, Int32, RSIMode)	RSI indicator
 	RSI(HistoricalData, Int32, RSIMode, PriceType)	RSI indicator
 	RVI(HistoricalData, Int32)	RVI indicator
 	RVI(HistoricalData, Int32, MAMode)	RVI indicator
 	SAR	SAR indicator

 	StdDev(FuncInt32, Double, Int32, MAMode)	StdDev indicator
 	StdDev(HistoricalData, Int32, MAMode, PriceType)	StdDev indicator
 	StdDev(HistoricalData, Int32, MAMode, Int32, PriceType)	StdDev indicator
 	Stochastic	Stochastic indicator
	ToString	Returns a string that represents the current object. (Inherited from Object)
 	WPR	WPR indicator

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicators Methods

The [BuiltInIndicators](#) type exposes the following members.

▲ Methods

Name	Description
AC	AC indicator
AD	AD indicator
ADX(HistoricalData, Int32)	ADX indicator
ADX(HistoricalData, Int32, MAMode)	ADX indicator
ADX(HistoricalData, Int32, MAMode, PriceType)	ADX indicator
Alligator	Alligator indicator
AO	AO indicator
ATR(HistoricalData, Int32)	ATR indicator
ATR(HistoricalData, Int32, MAMode)	ATR indicator
Bands(HistoricalData, Int32)	Bands indicator

 	Bands(HistoricalData, Int32, MAMode)	Bands indicator
 	Bands(HistoricalData, Int32, MAMode, Double)	Bands indicator
 	Bands(HistoricalData, FuncInt32, Double, Int32, MAMode, Double)	Bands indicator
 	Bands(HistoricalData, Int32, MAMode, Double, PriceType)	Bands indicator
 	BearsPower	BearsPower indicator
 	BullsPower	BullsPower indicator
 	BWMFI	BWMFI indicator
 	CCI(HistoricalData, Int32)	CCI indicator
 	CCI(HistoricalData, Int32, MAMode)	CCI indicator
 	CCI(HistoricalData, FuncInt32, Double, Int32, MAMode)	CCI indicator
 	CCI(HistoricalData, Int32, MAMode, PriceType)	CCI indicator
 	Custom	Custom indicator

	DeMarker	DeMarker indicator
	Envelopes(HistoricalData, FuncInt32, Double, Int32, MAMode, Double)	Envelopes indicator
	Envelopes(HistoricalData, Int32, MAMode, Double, PriceType)	Envelopes indicator
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	Force	Force indicator
	Fractals	Fractals indicator
	Gator	Gator indicator
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the

current
instance.
(Inherited
from Object)

	Ichimoku	Ichimoku indicator
	MA(FuncInt32, Double, Int32, MAMode)	MA indicator
	MA(HistoricalData, Int32, MAMode)	MA indicator
	MA(FuncInt32, Double, Int32, MAMode, Int32)	MA indicator
	MA(HistoricalData, Int32, MAMode, Int32, PriceType)	MA indicator
	MACD(HistoricalData, Int32, Int32, Int32)	MACD indicator
	MACD(HistoricalData, Int32, Int32, Int32, PriceType)	MACD indicator
	MFI	MFI indicator
	Momentum(FuncInt32, Double, Int32)	Momentum indicator
	Momentum(HistoricalData, Int32)	Momentum indicator
	Momentum(HistoricalData, Int32, PriceType)	Momentum indicator
	OBV	OBV

		indicator
	OsMA	OsMA indicator
	RSI(HistoricalData, Int32)	RSI indicator
	RSI(FuncInt32, Double, Int32, RSIMode)	RSI indicator
	RSI(HistoricalData, Int32, RSIMode)	RSI indicator
	RSI(HistoricalData, Int32, RSIMode, PriceType)	RSI indicator
	RVI(HistoricalData, Int32)	RVI indicator
	RVI(HistoricalData, Int32, MAMode)	RVI indicator
	SAR	SAR indicator
	StdDev(FuncInt32, Double, Int32, MAMode)	StdDev indicator
	StdDev(HistoricalData, Int32, MAMode, PriceType)	StdDev indicator
	StdDev(HistoricalData, Int32, MAMode, Int32, PriceType)	StdDev indicator
	Stochastic	Stochastic indicator
	ToString	Returns a

string that represents the current object.
(Inherited from [Object](#))



[WPR](#)

WPR
indicator

[Top](#)

◀ See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsAC Method

AC indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator AC(  
    HistoricalData data  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
BuiltInIndicatorsExample :  
IndicatorBuilder  
{  
    public  
    BuiltInIndicatorsExample()  
        : base()  
    {  
  
        Lines.Set("BuiltInIndicatorsEx  
ample");  
        #region  
        Initialization  
  
        Credentials.ProjectName =  
        "BuiltInIndicatorsExample";  
        #endregion  
    }  
  
    BuiltInIndicator  
    indicator;  
  
    public override  
    void Init()  
    {  
        indicator =  
        IndicatorsManager.BuildIn.AC(H  
istoryDataSeries);  
    }  
  
    public override  
    void Update(TickStatus args)  
    {  
        if  
(HistoryDataSeries.Count > 20)  
        {  
  
            indicator.GetValue();  
        }  
    }  
}
```

}

}

◀ See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsAD Method

AD indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator AD(  
    HistoricalData data  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
BuiltInIndicatorsExample :  
IndicatorBuilder  
{  
    public  
    BuiltInIndicatorsExample()  
        : base()  
    {  
  
        Lines.Set("BuiltInIndicatorsEx  
ample");  
        #region  
        Initialization  
  
        Credentials.ProjectName =  
        "BuiltInIndicatorsExample";  
        #endregion  
    }  
  
    BuiltInIndicator  
    indicator;  
  
    public override  
    void Init()  
    {  
        indicator =  
        IndicatorsManager.BuildIn.AD(H  
istoryDataSeries);  
    }  
  
    public override  
    void Update(TickStatus args)  
    {  
        if  
(HistoryDataSeries.Count > 20)  
        {  
  
            indicator.GetValue();  
        }  
    }  
}
```

}

}

◀ See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsADX Method

Overload List

Name	Description
 ADX(HistoricalData, Int32)	ADX indicator
 ADX(HistoricalData, Int32, MAMode)	ADX indicator
 ADX(HistoricalData, Int32, MAMode, PriceType)	ADX indicator

[Top](#)

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsADX(Historical Data, Int32) Method

ADX indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator ADX(Copy
    HistoricalData data,
    int period
)
```

Parameters

data [HistoricalData](#)

 indicator historical data

period [Int32](#)

 indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExampl
e");
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }

                BuiltInIndicator
indicator;

        public override void
Init()
        {
            indicator =
IndicatorsManager.BuildIn.ADX(Hist
oryDataSeries, 12);
        }

        public override void
Update(TickStatus args)
        {
            if
(HistoryDataSeries.Count > 20)
```

```
        {  
            indicator.GetValue();  
        }  
    }  
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[ADX Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsADX(Historical Data, Int32, MAMode) Method

ADX indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator ADX(  
    HistoricalData data,  
    int period,  
    MAMode mode  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
    #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
    #endregion
}

    BuiltInIndicator
indicator;

    public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.ADX(
HistoryDataSeries, 12,
MAMode.EMA);
    }
}
```

Copy

```
        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[ADX Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsADX(Historical Data, Int32, MAMode, PriceType) Method

ADX indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

▪ Syntax

C#

```
public BuiltInIndicator ADX(  
    HistoricalData data,  
    int period,  
    MAMode mode,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode
priceType [PriceType](#)
 indicator price type

Return Value BuiltInIndicator

Example

C#

Copy

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.ADX(
```

```
HistoryDataSeries, 12,
MAMode.EMA, PriceType.Close);
}

public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[ADX Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsAlligator Method

Alligator indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator
Alligator(
    HistoricalData data,
    int jaw_period,
    int jaw_shift,
    int teeth_period,
    int teeth_shift,
    int lips_period,
    int lips_shift,
    MAMode mode,
    PriceType priceType
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data
jaw_period [Int32](#)
indicator jaw period
jaw_shift [Int32](#)

```
    indicator jaw shift
teeth_period Int32
    indicator teeth period
teeth_shift Int32
    indicator teeth shift
lips_period Int32
    indicator lips period
lips_shift Int32
    indicator lips shift
mode MAMode
    indicator ma mode
priceType PriceType
    indicator price type
```

Return Value

[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.History;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsEx
ample");
            #region
Initialization
```

[Copy](#)

```
Credentials.ProjectName =
"BuiltInIndicatorsExample";
#endregion
}

        BuiltInIndicator
indicator;

    public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.Alli-
gator(HistoryDataSeries, 12,
0, 10, 0, 6, 0, MAMode.EMA,
PriceType.Close);
}

    public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

    indicator.GetValue();
}
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsAO Method

AO indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator AO(  
    HistoricalData data  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
BuiltInIndicatorsExample :  
IndicatorBuilder  
{  
    public  
    BuiltInIndicatorsExample()  
        : base()  
    {  
  
        Lines.Set("BuiltInIndicatorsEx  
ample");  
        #region  
        Initialization  
  
        Credentials.ProjectName =  
        "BuiltInIndicatorsExample";  
        #endregion  
    }  
  
    BuiltInIndicator  
    indicator;  
  
    public override  
    void Init()  
    {  
        indicator =  
        IndicatorsManager.BuildIn.AO(H  
istoryDataSeries);  
    }  
  
    public override  
    void Update(TickStatus args)  
    {  
        if  
(HistoryDataSeries.Count > 20)  
        {  
  
            indicator.GetValue();  
        }  
    }  
}
```

}

}

◀ See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicators.ATR Method

Overload List

Name	Description
  ATR(HistoricalData, Int32)	ATR indicator
  ATR(HistoricalData, Int32, MAMode)	ATR indicator

[Top](#)

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsATR(Historical Data, Int32) Method

ATR indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator ATR(Copy
    HistoricalData data,
    int period
)
```

Parameters

data [HistoricalData](#)

 indicator historical data

period [Int32](#)

 indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExampl
e");
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }
    }

        BuiltInIndicator
indicator;

        public override void
Init()
    {
        indicator =
IndicatorsManager.BuildIn.ATR(Hist
oryDataSeries, 12);
    }

        public override void
Update(TickStatus args)
    {
        if
(HistoryDataSeries.Count > 20)
```

```
        {  
            indicator.GetValue();  
        }  
    }  
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[ATR Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsATR(Historical Data, Int32, MAMode) Method

ATR indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator ATR(  
    HistoricalData data,  
    int period,  
    MAMode mode  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
    #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
    #endregion
}

    BuiltInIndicator
indicator;

    public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.ATR(
HistoryDataSeries, 12,
MAMode.EMA);
    }
}
```

Copy

```
        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[ATR Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsBands Method

Overload List

Name	Description
 Bands(HistoricalData, Int32)	Bands indicator
 Bands(HistoricalData, Int32, MAMode)	Bands indicator
 Bands(HistoricalData, Int32, MAMode, Double)	Bands indicator
 Bands(HistoricalData, FuncInt32, Double, Int32, MAMode, Double)	Bands indicator
 Bands(HistoricalData, Int32, MAMode, Double, PriceType)	Bands indicator

[Top](#)

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsBands(Historical Data, Int32) Method

Bands indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator Bands(  
    HistoricalData data,  
    int period  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExample");
                #region Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }

                BuiltInIndicator
indicator;

        public override void
Init()
        {
            indicator =
IndicatorsManager.BuildIn.Bands(HistoryDataSeries, 12);
        }

        public override void
Update(TickStatus args)
        {
            if
(HistoryDataSeries.Count > 20)
```

```
        {  
            indicator.GetValue();  
        }  
    }  
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[Bands Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsBands(Historical Data, Int32, MAMode) Method

Bands indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator Bands ( Copy
    HistoricalData data,
    int period,
    MAMode mode
)
```

Parameters

data [HistoricalData](#)
indicator historical data
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
    #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
    #endregion
}

    BuiltInIndicator
indicator;

    public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.Band
s(HistoryDataSeries, 12,
MAMode.EMA);
    }
}
```

Copy

```
        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[Bands Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsBands(Historical Data, Int32, MAMode, Double) Method

Bands indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator Bands ( Copy
    HistoricalData data,
    int period,
    MAMode mode,
    double deviation
)
```

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode
deviation [Double](#)
 indicator deviation

Return Value BuiltInIndicator

Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.Band
s(HistoryDataSeries, 12,
```

```
        MAMode.EMA, 0.02);
    }

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
    {

indicator.GetValue();
    }
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[Bands Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsBands(Historical Data, FuncInt32, Double, Int32, MAMode, Double) Method

Bands indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator Bands(Copy
    HistoricalData data,
    Func<int, double>
    func,
    int period,
    MAMode mode,
    double deviation
)
```

Parameters

data [HistoricalData](#)
indicator historical data
func [FuncInt32, Double](#)
indicator func delegate
period [Int32](#)

```
    indicator period
mode MAMode
    indicator ma mode
deviation Double
    indicator deviation
```

Return Value
BuiltInIndicator

Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;
```

```
        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.Bands(HistoryDataSeries, (int
index) => { return
HistoryDataSeries.GetValue(PriceType.Close, index); }, 6,
MAMode.EMA, 0.02);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[Bands Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsBands(Historical Data, Int32, MAMode, Double, PriceType) Method

Bands indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator Bands(Copy
    HistoricalData data,
    int period,
    MAMode mode,
    double deviation,
    PriceType priceType
)
```

Parameters

data [HistoricalData](#)
indicator historical data
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode

```
deviation Double  
    indicator deviation  
priceType PriceType  
    indicator price type
```

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
BuiltInIndicatorsExample :  
IndicatorBuilder  
    {  
        public  
BuiltInIndicatorsExample()  
            : base()  
        {  
  
Lines.Set("BuiltInIndicatorsEx  
ample");  
            #region  
Initialization  
  
Credentials.ProjectName =  
"BuiltInIndicatorsExample";  
            #endregion  
        }  
  
        BuiltInIndicator  
indicator;
```

Copy

```
        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.Band
s(HistoryDataSeries, 12,
MAMode.EMA, 0.02,
PriceType.Close);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[Bands Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsBears Power Method

BearsPower indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator  
BearsPower(  
    HistoricalData data,  
    int period,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
priceType [PriceType](#)
 indicator price type

Return Value

[BuiltInIndicator](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
        : base()
        {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
        {
            indicator =
IndicatorsManager.BuildIn.Bear
sPower(HistoryDataSeries, 12,
PriceType.Close);
        }

        public override
void Update(TickStatus args)
        {
```

```
        if  
    (HistoryDataSeries.Count > 20)  
    {  
  
        indicator.GetValue();  
    }  
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsBulls Power Method

BullsPower indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator  
BullsPower(  
    HistoricalData data,  
    int period,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
priceType [PriceType](#)
 indicator price type

Return Value

[BuiltInIndicator](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
        : base()
        {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
        {
            indicator =
IndicatorsManager.BuildIn.Bull
sPower(HistoryDataSeries, 12,
PriceType.Close);
        }

        public override
void Update(TickStatus args)
        {
```

```
        if  
    (HistoryDataSeries.Count > 20)  
    {  
  
        indicator.GetValue();  
    }  
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsBWMFI Method

BWMFI indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator BWMFI(  
    HistoricalData data  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data

Return Value
[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv
```

[Copy](#)

```
    {
        public class
BuiltInIndicatorsExample : IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
: base()
{

Lines.Set("BuiltInIndicatorsEx
ample");
#region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
#endregion
}

        BuiltInIndicator
indicator;

        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.BWMF
I(HistoryDataSeries);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsCCI Method

Overload List

Name	Description
 CCI(HistoricalData, Int32)	CCI indicator
 CCI(HistoricalData, Int32, MAMode)	CCI indicator
 CCI(HistoricalData, FuncInt32, Double, Int32, MAMode)	CCI indicator
 CCI(HistoricalData, Int32, MAMode, PriceType)	CCI indicator

[Top](#)

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsCCI(Historical Data, Int32) Method

CCI indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator CCI(Copy
    HistoricalData data,
    int period
)
```

Parameters

data [HistoricalData](#)

 indicator historical data

period [Int32](#)

 indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExampl
e");
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }
    }

        BuiltInIndicator
indicator;

        public override void
Init()
    {
        indicator =
IndicatorsManager.BuildIn.CCI(Hist
oryDataSeries, 12);
    }

        public override void
Update(TickStatus args)
    {
        if
(HistoryDataSeries.Count > 20)
```

```
        {  
            indicator.GetValue();  
        }  
    }  
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[CCI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsCCI(Historical Data, Int32, MAMode) Method

CCI indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator CCI(  
    HistoricalData data,  
    int period,  
    MAMode mode  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
    #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
    #endregion
}

    BuiltInIndicator
indicator;

    public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.CCI(
HistoryDataSeries, 12,
MAMode.EMA);
    }
}
```

Copy

```
        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[CCI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsCCI(Historical Data, FuncInt32, Double, Int32, MAMode) Method

CCI indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator CCI(  
    HistoricalData data,  
    Func<int, double>  
    func,  
    int period,  
    MAMode mode  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
func [FuncInt32, Double](#)
 indicator func delegate
period [Int32](#)
 indicator period

```
mode MAMode  
    indicator ma mode
```

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
BuiltInIndicatorsExample :  
IndicatorBuilder  
    {  
        public  
BuiltInIndicatorsExample()  
            : base()  
        {  
  
Lines.Set("BuiltInIndicatorsEx  
ample");  
            #region  
Initialization  
  
Credentials.ProjectName =  
"BuiltInIndicatorsExample";  
            #endregion  
    }  
  
        BuiltInIndicator  
indicator;  
  
        public override  
void Init()
```

Copy

```
        {
            indicator =
IndicatorsManager.BuildIn.CCI(
HistoryDataSeries, (int index)
=> { return
HistoryDataSeries.GetValue(Pri
ceType.Close, index); }, 6,
MAMode.EMA);
        }

    public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[CCI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsCCI(Historical Data, Int32, MAMode, PriceType) Method

CCI indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator CCI(  
    HistoricalData data,  
    int period,  
    MAMode mode,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode
priceType [PriceType](#)
 indicator price type

Return Value BuiltInIndicator

Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.History;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.CCI(
```

```
HistoryDataSeries, 12,
MAMode.EMA, PriceType.Close);
}

public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[CCI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsCustom Method

Custom indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator
Custom(
    string name,
    HistoricalData data,
    params Object[]
arguments
)
```

[Copy](#)

Parameters

name [String](#)
indicator name
data [HistoricalData](#)
indicator historical data
arguments [Object](#)
indicator arguments

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
    #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
    #endregion
}

    BuiltInIndicator
indicator;

    public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.Cust
om("Custom",
HistoryDataSeries);
    }
}
```

Copy

```
        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsDeMarker Method

DeMarker indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator  
DeMarker(  
    HistoricalData data,  
    int period  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;
```

[Copy](#)

```
namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.DeMa
rker(HistoryDataSeries, 12);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{
```

```
        indicator.GetValue();
    }
}
```

▲ See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsEnvelopes Method

▪ Overload List

Name	Description
<code>Envelopes(HistoricalData, FuncInt32, Double, Int32, MAMode, Double)</code>	Envelopes indicator
<code>Envelopes(HistoricalData, Int32, MAMode, Double, PriceType)</code>	Envelopes indicator

[Top](#)

▪ See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsEnvelopes(Historical Data, FuncInt32, Double, Int 32, MAMode, Double) Method

Envelopes indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public BuiltInIndicator Envelopes(Copy
    HistoricalData data,
    Func<int, double> func,
    int period,
    MAMode mode,
    double deviation
)
```

Parameters

data [HistoricalData](#)
indicator historical data
func [FuncInt32, Double](#)
indicator func delegate
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode
deviation [Double](#)
indicator period

Return Value

[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :  
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
        : base()
        {

Lines.Set("BuiltInIndicatorsExample"
);
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }
    }

        BuiltInIndicator
indicator;
    public override void
Init()
    {
        indicator =
IndicatorsManager.BuildIn.Envelopes(
HistoryDataSeries, (int index) => {
return
HistoryDataSeries.GetValue(PriceType
.Close, index); }, 6, MAMode.EMA,
0.02);
    }

    public override void
```

Copy

```
Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
    {

indicator.GetValue();
    }
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[Envelopes Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsEnvelopes(Historical Data, Int32, MAMode, Double, PriceType) Method

Envelopes indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public BuiltInIndicator Envelopes(Copy
    HistoricalData data,
    int period,
    MAMode mode,
    double deviation,
    PriceType priceType
)
```

Parameters

data [HistoricalData](#)
indicator historical data
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode
deviation [Double](#)
indicator deviation
priceType [PriceType](#)
indicator price type

Return Value

[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :  
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
        : base()
        {

Lines.Set("BuiltInIndicatorsExample"
);
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }
    }

        BuiltInIndicator
indicator;

        public override void
Init()
        {
            indicator =
IndicatorsManager.BuildIn.Envelopes(
HistoryDataSeries, 12, MAMode.EMA,
0.02, PriceType.Close);
        }

        public override void
Update(TickStatus args)
        {
            if
```

Copy

```
(HistoryDataSeries.Count > 20)
{
    indicator.GetValue();
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[Envelopes Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsForce Method

Force indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator Force(Copy
    HistoricalData data,
    int period,
    MAMode mode,
    PriceType priceType
)
```

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode
priceType [PriceType](#)
 indicator price type

Return Value

[BuiltInIndicator](#)

► Example

C#[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.History;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :  
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsEx  
ample");
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }

                BuiltInIndicator
indicator;
public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.Forc  
e(HistoryDataSeries, 12,
MAMode.EMA, PriceType.Close);
}

public override
```

```
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
    {

indicator.GetValue();
    }
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsFractals Method

Fractals indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator
Fractals(
    HistoricalData data
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data

Return Value
[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
```

[Copy](#)

```
namespace TestEnv
{
    public class BuiltInIndicatorsExample : IndicatorBuilder
    {
        public BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExample");
#region Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
#endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.Frac
tals(HistoryDataSeries);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
```

```
        }  
    }  
}
```

◀ See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsGator Method

Gator indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator Gator(Copy
    HistoricalData data,
    int jaw_period,
    int jaw_shift,
    int teeth_period,
    int teeth_shift,
    int lips_period,
    int lips_shift,
    MAMode mode,
    PriceType priceType
)
```

Parameters

data [HistoricalData](#)
indicator historical data

jaw_period [Int32](#)
indicator jaw period

jaw_shift [Int32](#)
indicator jaw shift

teeth_period [Int32](#)
indicator teeth period

```
teeth_shift Int32  
    indicator teeth shift  
lips_period Int32  
    indicator lips period  
lips_shift Int32  
    indicator lips shift  
mode MAMode  
    indicator ma mode  
priceType PriceType  
    indicator price type
```

Return Value

[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
BuiltInIndicatorsExample :  
IndicatorBuilder  
    {  
        public  
BuiltInIndicatorsExample()  
            : base()  
        {  
  
Lines.Set("BuiltInIndicatorsEx  
ample");  
            #region  
Initialization  
  
Credentials.ProjectName =  
"BuiltInIndicatorsExample";
```

[Copy](#)

```
        #endregion
    }

    BuiltInIndicator
indicator;

        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.Gato
r(HistoryDataSeries, 12, 0,
10, 0, 6, 0, MAMode.EMA,
PriceType.Close);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

    indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsIchimoku Method

Ichimoku indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator
Ichimoku(
    HistoricalData data,
    int tenkan_sen,
    int kijun_sen,
    int senkou_span
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
tenkan_sen [Int32](#)
 indicator tenkan sen
kijun_sen [Int32](#)
 indicator kijun sen
senkou_span [Int32](#)
 indicator senkou span

Return Value

BuiltInIndicator

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsEx
ample");
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }

            BuiltInIndicator
indicator;

        public override
void Init()
        {
            indicator =
IndicatorsManager.BuildIn.Ichi
moku(HistoryDataSeries, 24,
18, 34);
    }
```

Copy

```
        }

    public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

    indicator.GetValue();
}
}

}
```

See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsMA Method

Overload List

Name	Description
 MA(FuncInt32, Double, Int32, MAMode)	MA indicator
 MA(HistoricalData, Int32, MAMode)	MA indicator
 MA(FuncInt32, Double, Int32, MAMode, Int32)	MA indicator
 MA(HistoricalData, Int32, MAMode, Int32, PriceType)	MA indicator

[Top](#)

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMA(FuncInt32, Double, Int32, MAMode) Method

MA indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator MA(Copy
    Func<int, double>
    func,
    int period,
    MAMode mode
)
```

Parameters

func [FuncInt32, Double](#)
indicator func delegate

period [Int32](#)
indicator period

mode [MAMode](#)
indicator ma mode

Return Value

[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExample");
            #region Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
            #endregion
        }

            BuiltInIndicator
indicator;

        public override
void Init()
        {
            indicator =
IndicatorsManager.BuildIn.MA( (
int index) => { return
HistoryDataSeries.GetValue(Pri
ceType.Close, index); }, 6,
```

Copy

```
        MAMode.EMA) ;
    }

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
    {

indicator.GetValue();
    }
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[MA Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMA(Historical Data, Int32, MAMode) Method

MA indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator MA(  
    HistoricalData data,  
    int period,  
    MAMode mode  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
    #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
    #endregion
}

    BuiltInIndicator
indicator;

    public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 12,
MAMode.EMA);
    }
}
```

Copy

```
        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[MA Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMA(FuncInt 32, Double, Int32, MAMode, Int32) Method

MA indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator MA(  
    Func<int, double>  
    func,  
    int period,  
    MAMode mode,  
    int ma_shift  
)
```

[Copy](#)

Parameters

func [FuncInt32, Double](#)
indicator func delegate
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode

```
ma_shift Int32  
    indicator ma shift
```

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
BuiltInIndicatorsExample :  
IndicatorBuilder  
    {  
        public  
BuiltInIndicatorsExample()  
            : base()  
        {  
  
Lines.Set("BuiltInIndicatorsEx  
ample");  
            #region  
Initialization  
  
Credentials.ProjectName =  
"BuiltInIndicatorsExample";  
            #endregion  
    }  
  
        BuiltInIndicator  
indicator;  
  
        public override  
void Init()
```

Copy

```
        {
            indicator =
IndicatorsManager.BuildIn.MA( (
int index) => { return
HistoryDataSeries.GetValue(Pri
ceType.Close, index); }, 6,
MAMode.EMA, 1);
        }

    public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{
    indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[MA Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMA(Historical Data, Int32, MAMode, Int32, PriceType) Method

MA indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator MA(  
    HistoricalData data,  
    int period,  
    MAMode mode,  
    int ma_shift,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode

```
ma_shift Int32  
    indicator ma shift  
priceType PriceType  
    indicator price type
```

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
BuiltInIndicatorsExample :  
IndicatorBuilder  
    {  
        public  
BuiltInIndicatorsExample()  
            : base()  
        {  
  
Lines.Set("BuiltInIndicatorsEx  
ample");  
            #region  
Initialization  
  
Credentials.ProjectName =  
"BuiltInIndicatorsExample";  
            #endregion  
        }  
  
        BuiltInIndicator  
indicator;
```

Copy

```
        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 12,
MAMode.EMA, 1,
PriceType.Close);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[MA Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsMACD Method

Overload List

Name	Description
 MACD(HistoricalData, Int32, Int32, Int32)	MACD indicator
 MACD(HistoricalData, Int32, Int32, Int32, PriceType)	MACD indicator

[Top](#)

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMACD(Historical Data, Int32, Int32, Int32) Method

MACD indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public BuiltInIndicator MACD(Copy
    HistoricalData data,
    int fast_ema_period,
    int slow_ema_period,
    int signal_period
)
```

Parameters

data [HistoricalData](#)
 indicator historical data
fast_ema_period [Int32](#)
 indicator fast ema period
slow_ema_period [Int32](#)
 indicator slow ema period
signal_period [Int32](#)
 indicator signal period

Return Value BuiltInIndicator

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.MACD
(HistoryDataSeries, 24, 18,
```

Copy

```
    34) ;
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[MACD Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMACD(Historical Data, Int32, Int32, Int32, PriceType) Method

MACD indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator MACD(  
    HistoricalData data,  
    int fast_ema_period,  
    int slow_ema_period,  
    int signal_period,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
fast_ema_period [Int32](#)
 indicator fast ema period
slow_ema_period [Int32](#)
 indicator slow ema period
signal_period [Int32](#)
 indicator signal period

priceType [PriceType](#)
indicator price type

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
```

```
        {
            indicator =
IndicatorsManager.BuildIn.MACD
(HistoryDataSeries, 24, 18,
34, PriceType.Close);
        }

    public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[MACD Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsMFI Method

MFI indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator MFI(  
    HistoricalData data,  
    int period  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data

period [Int32](#)
indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;
```

[Copy](#)

```
namespace TestEnv
{
    public class BuiltInIndicatorsExample : IndicatorBuilder
    {
        public BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExample");
#region Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
#endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.MFI(
HistoryDataSeries, 24);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
```

```
        }  
    }  
}
```

◀ See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMomentum Method

▪ Overload List

Name	Description
 Momentum(FuncInt32, Double, Int32)	Momentum indicator
 Momentum(HistoricalData, Int32)	Momentum indicator
 Momentum(HistoricalData, Int32, PriceType)	Momentum indicator

[Top](#)

▪ See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMomentum(FuncInt 32, Double, Int32) Method

Momentum indicator

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator Momentum(  
    Func<int, double> func,  
    int period  
)
```

Copy

Parameters

func FuncInt32, Double
indicator func delegate
period Int32
indicator period

Return Value

BuiltInIndicator

► Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{
```

Copy

```
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
: base()
{

Lines.Set("BuiltInIndicatorsExample
");
#region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
#endregion
}

        BuiltInIndicator
indicator;

        public override void
Init()
{
    indicator =
IndicatorsManager.BuildIn.Momentum(
(int index) => { return
HistoryDataSeries.GetValue(PriceTyp
e.Close, index); }, 6);
}

        public override void
Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[Momentum Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMomentum(Historical Data, Int32) Method

Momentum indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator Momentum(  
    HistoricalData data,  
    int period  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
BuiltInIndicatorsExample :  
IndicatorBuilder  
{
```

[Copy](#)

```
    public
    BuiltInIndicatorsExample()
        : base()
        {

    Lines.Set("BuiltInIndicatorsExample");
                #region Initialization

    Credentials.ProjectName =
    "BuiltInIndicatorsExample";
                #endregion
        }

    BuiltInIndicator
    indicator;

    public override void
    Init()
    {
        indicator =
    IndicatorsManager.BuildIn.Momentum(His
    toryDataSeries, 24);
    }

    public override void
    Update(TickStatus args)
    {
        if
    (HistoryDataSeries.Count > 20)
        {

    indicator.GetValue();
        }
    }
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[Momentum Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsMomentum(Historical Data, Int32, PriceType) Method

Momentum indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator Momentum(  
    HistoricalData data,  
    int period,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
priceType [PriceType](#)
 indicator price type

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using TradeApi.History;  
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
: base()
{

Lines.Set("BuiltInIndicatorsExample");
#region Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
#endregion
}

    BuiltInIndicator
indicator;

    public override void
Init()
{
    indicator =
IndicatorsManager.BuildIn.Momentum(His
toryDataSeries, 24, PriceType.Close);
}

    public override void
Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{
    indicator.GetValue();
}
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

Momentum Overload
TradeApi.Indicators Namespace

BuiltInIndicatorsOBV Method

OBV indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator OBV(Copy
    HistoricalData data,
    PriceType priceType
)
```

Parameters

data [HistoricalData](#)
indicator historical data
priceType [PriceType](#)
indicator price type

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;
```

```
namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.OBV(
HistoryDataSeries,
PriceType.Close);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{
```

```
        indicator.GetValue();
    }
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsOsMA Method

OsMA indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator OsMA(  
    HistoricalData data,  
    int fast_ema_period,  
    int slow_ema_period,  
    int signal_period,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)

 indicator historical data

fast_ema_period [Int32](#)

 indicator fast ema period

slow_ema_period [Int32](#)

 indicator slow ema period

signal_period [Int32](#)

 indicator signal period

priceType [PriceType](#)

 indicator price type

Return Value

BuiltInIndicator

Example

C#

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.OsMA
(HistoryDataSeries, 6, 24, 8,
```

Copy

```
    PriceType.Close);
}

public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
    {

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsRSI Method

Overload List

Name	Description
 RSI(HistoricalData, Int32)	RSI indicator
 RSI(FuncInt32, Double, Int32, RSIMode)	RSI indicator
 RSI(HistoricalData, Int32, RSIMode)	RSI indicator
 RSI(HistoricalData, Int32, RSIMode, PriceType)	RSI indicator

[Top](#)

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsRSI(Historical Data, Int32) Method

RSI indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator RSI(Copy
    HistoricalData data,
    int period
)
```

Parameters

data [HistoricalData](#)

 indicator historical data

period [Int32](#)

 indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExampl
e");
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }
    }

        BuiltInIndicator
indicator;

        public override void
Init()
    {
        indicator =
IndicatorsManager.BuildIn.RSI(Hist
oryDataSeries, 6);
    }

        public override void
Update(TickStatus args)
    {
        if
(HistoryDataSeries.Count > 20)
```

```
        {  
            indicator.GetValue();  
        }  
    }  
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[RSI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsRSI(FuncInt 32, Double, Int32, RSIMode) Method

RSI indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator RSI(  
    Func<int, double>  
    func,  
    int period,  
    RSIMode mode  
)
```

[Copy](#)

Parameters

func [FuncInt32, Double](#)
 indicator func delegate
period [Int32](#)
 indicator period
mode [RSIMode](#)
 indicator rsi mode

Return Value

[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExample");
            #region Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
            #endregion
        }

            BuiltInIndicator
indicator;

            public override
void Init()
        {
            indicator =
IndicatorsManager.BuildIn.RSI(
(int index) => { return
HistoryDataSeries.GetValue(Pri
ceType.Close, index); }, 6,
```

Copy

```
        RSIMode.Simple);
    }

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{
    indicator.GetValue();
}
}

}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[RSI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsRSI(Historical Data, Int32, RSIMode) Method

RSI indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator RSI(  
    HistoricalData data,  
    int period,  
    RSIMode mode  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [RSIMode](#)
 indicator rsi mode

Return Value

[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
    #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
    #endregion
}

    BuiltInIndicator
indicator;

    public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.RSI(
HistoryDataSeries, 6,
RSIMode.Simple);
    }
}
```

Copy

```
        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[RSI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsRSI(Historical Data, Int32, RSIMode, PriceType) Method

RSI indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator RSI(  
    HistoricalData data,  
    int period,  
    RSIMode mode,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [RSIMode](#)
 indicator rsi mode
priceType [PriceType](#)
 indicator price type

Return Value BuiltInIndicator

Example

C#

Copy

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.RSI(
```

```
    HistoryDataSeries, 6,
    RSIMode.Simple,
    PriceType.Close);
}

public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{
    indicator.GetValue();
}
}

}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[RSI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicators.RVI Method

Overload List

Name	Description
  RVI(HistoricalData, Int32)	RVI indicator
  RVI(HistoricalData, Int32, MAMode)	RVI indicator

[Top](#)

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltIn Indicators.RVI(Historical Data, Int32) Method

RVi indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator RVI(Copy
    HistoricalData data,
    int period
)
```

Parameters

data [HistoricalData](#)

 indicator historical data

period [Int32](#)

 indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExampl
e");
                #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
                #endregion
        }
    }

        BuiltInIndicator
indicator;

        public override void
Init()
    {
        indicator =
IndicatorsManager.BuildIn.RVI(Hist
oryDataSeries, 6);
    }

        public override void
Update(TickStatus args)
    {
        if
(HistoryDataSeries.Count > 20)
```

```
        {  
            indicator.GetValue();  
        }  
    }  
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[RVI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsRVI(Historical Data, Int32, MAMode) Method

RVI indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator RVI(  
    HistoricalData data,  
    int period,  
    MAMode mode  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.RVI(
HistoryDataSeries, 6,
MAMode.EMA);
    }
}
```

Copy

```
        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[RVI Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsSAR Method

SAR indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator SAR(  
    HistoricalData data,  
    double step,  
    double koefficient  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
indicator historical data

step [Double](#)
indicator step

koefficient [Double](#)
indicator koefficient

Return Value

[BuiltInIndicator](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : IndicatorBuilder
    {
        public
BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsEx
ample");
            #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
            #endregion
        }
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.RVI(
HistoryDataSeries, 6,
MAMode.EMA);
    }

        public override
void Update(TickStatus args)
    {
        if
```

```
(HistoryDataSeries.Count > 20)
{
    indicator.GetValue();
}
}
```

◀ See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsStd Dev Method

▪ Overload List

Name	Description
 StdDev(FuncInt32, Double, Int32, MAMode)	StdDev indicator
 StdDev(HistoricalData, Int32, MAMode, PriceType)	StdDev indicator
 StdDev(HistoricalData, Int32, MAMode, Int32, PriceType)	StdDev indicator

[Top](#)

▪ See Also

[Reference](#)

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsStd Dev(FuncInt32, Double, Int32, MAMode) Method

StdDev indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator  
StdDev(  
        Func<int, double>  
        func,  
        int period,  
        MAMode mode  
)
```

[Copy](#)

Parameters

func [FuncInt32, Double](#)
indicator func delegate
period [Int32](#)
indicator period
mode [MAMode](#)
indicator ma mode

Return Value

BuiltInIndicator

Example

C#

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample :
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
    : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.StdD
ev((int index) => { return
```

Copy

```
    HistoryDataSeries.GetValue(Pri  
ceType.Close, index); }, 6,  
MAMode.EMA);  
}  
  
        public override  
void Update(TickStatus args)  
{  
    if  
(HistoryDataSeries.Count > 20)  
    {  
  
        indicator.GetValue();  
    }  
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[StdDev Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsStd Dev(HistoricalData, Int32, MAMode, Price Type) Method

StdDev indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator  
StdDev(  
    HistoricalData data,  
    int period,  
    MAMode mode,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode
priceType [PriceType](#)
 indicator price type

Return Value BuiltInIndicator

Example

C#

Copy

```
using Runtime.Script;
using TradeApi.History;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
    {
        indicator =
IndicatorsManager.BuildIn.StdD
```

```
ev(HistoryDataSeries, 6,
MAMode.EMA, PriceType.Close);
}

public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

[Reference](#)

[BuiltInIndicators Class](#)

[StdDev Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsStd Dev(HistoricalData, Int32, MAMode, Int 32, PriceType) Method

StdDev indicator

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator  
StdDev(  
    HistoricalData data,  
    int period,  
    MAMode mode,  
    int ma_shift,  
    PriceType priceType  
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data
period [Int32](#)
 indicator period
mode [MAMode](#)
 indicator ma mode
ma_shift [Int32](#)

```
    indicator ma shift  
priceType PriceType  
    indicator price type
```

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
BuiltInIndicatorsExample :  
IndicatorBuilder  
    {  
        public  
BuiltInIndicatorsExample()  
            : base()  
            {  
  
Lines.Set("BuiltInIndicatorsEx  
ample");  
                #region  
Initialization  
  
Credentials.ProjectName =  
"BuiltInIndicatorsExample";  
                #endregion  
            }  
  
            BuiltInIndicator  
indicator;  
  
        public override
```

Copy

```
void Init()
{
    indicator =
IndicatorsManager.BuildIn.StdD
ev(HistoryDataSeries, 6,
MAMode.EMA, 1,
PriceType.Close);
}

public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[StdDev Overload](#)

[TradeApi.Indicators Namespace](#)

BuiltIn IndicatorsStochastic Method

Stochastic indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator
Stochastic(
    HistoricalData data,
    int kperiod,
    int dperiod,
    int slowing,
    MAMode mode
)
```

[Copy](#)

Parameters

data [HistoricalData](#)
 indicator historical data

kperiod [Int32](#)
 indicator k period

dperiod [Int32](#)
 indicator d period

slowing [Int32](#)
 indicatorslowing

mode [MAMode](#)

indicator ma mode

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BuiltInIndicatorsExample : 
IndicatorBuilder
{
    public
BuiltInIndicatorsExample()
        : base()
    {

Lines.Set("BuiltInIndicatorsEx
ample");
        #region
Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
        #endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
{
    indicator =

```

Copy

```
IndicatorsManager.BuildIn.Stochastic(HistoryDataSeries, 8,
12, 6, MAMode.EMA);
}

public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
}
}
```

See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInIndicatorsWPR Method

WPR indicator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator WPR(Copy
    HistoricalData data,
    int period
)
```

Parameters

data [HistoricalData](#)
indicator historical data
period [Int32](#)
indicator period

Return Value

[BuiltInIndicator](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
```

```
namespace TestEnv
{
    public class BuiltInIndicatorsExample : IndicatorBuilder
    {
        public BuiltInIndicatorsExample()
            : base()
        {

Lines.Set("BuiltInIndicatorsExample");
#region Initialization

Credentials.ProjectName =
"BuiltInIndicatorsExample";
#endregion
    }

        BuiltInIndicator
indicator;

        public override
void Init()
{
    indicator =
IndicatorsManager.BuildIn.WPR(
HistoryDataSeries, 6);
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 20)
{

indicator.GetValue();
}
```

```
        }  
    }  
}
```

◀ See Also

Reference

[BuiltInIndicators Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInLine Class

Built in line entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.Indicators](#)
[BuiltInLine](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
    BuiltInLine
```

[Copy](#)

The [BuiltInLine](#) type exposes the following members.

▪ Properties

	Name	Description
 	Item	Returns build in indicator line value by index

[Top](#)

▪ Methods

	Name	Description
--	------	-------------

	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	GetValue	Returns build in indicator value by bar offset
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

Reference

[TradeApi.Indicators Namespace](#)

BuiltInLine Properties

The [BuiltInLine](#) type exposes the following members.

► Properties

	Name	Description
 	Item	Returns build in indicator line value by index

[Top](#)

► See Also

[Reference](#)

[BuiltInLine Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInLineItem Property

Returns build in indicator line value by index

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double this[  
    int offset  
] { get; }
```

[Copy](#)

Parameters

offset [Int32](#)

The element offset in the historical series

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
IndexerExample :
IndicatorBuilder
{
    public
LinesExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"IndexerExample";
        #endregion

        Lines.Set("IndexerExample");

        base.SeparateWindow = false;
    }

        BuiltInIndicator
buildIn;

        public override
void Init()
{
    buildIn =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 2,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{
    if(args ==
TickStatus.IsQuote)
        buildIn[0][0];
}
```

```
        else
            buildIn[0][1];
        }
    }
```

◀ See Also

[Reference](#)

[BuiltInLine Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInLine Methods

The [BuiltInLine](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	GetValue	Returns build in indicator value by bar offset
	ToString	Returns a string that represents the

current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BuiltInLine Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInLineGetValue Method

Returns build in indicator value by bar offset

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetValue(Copy
                      int offset = 0
)
```

Parameters

offset [Int32](#) (Optional)

The element offset in the historical series

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{Copy
```

```
    public class
GetValueExample :
IndicatorBuilder
{
    public
GetValueExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"GetValueExample";
    #endregion

Lines.Set("GetValueExample");

base.SeparateWindow = false;
}

        BuiltInIndicator
buildIn;

    public override
void Init()
{
    buildIn =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 2,
MAMode.EMA);
}

    public override
void Update(TickStatus args)
{
    if(args ==
TickStatus.IsQuote)
```

```
buildIn.GetValue(offset: 0);  
    else  
  
    buildIn.GetValue(offset: 1);  
        }  
    }  
}
```

See Also

Reference

[BuiltInLine Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInLines Class

Built in lines entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.Indicators](#)
[BuiltInLines](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
    BuiltInLines
```

[Copy](#)

The [BuiltInLines](#) type exposes the following members.

▪ Properties

	Name	Description
	Count	Gets indicator's values count
	Item	Returns build in indicator line object by index

[Top](#)

◀ Methods

	Name	Description
≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetType	Gets the Type of the current instance. (Inherited from Object)
≡	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

BuiltInLines Properties

The [BuiltInLines](#) type exposes the following members.

► Properties

	Name	Description
	Count	Gets indicator's values count
	Item	Returns build in indicator line object by index

[Top](#)

► See Also

[Reference](#)

[BuiltInLines Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInLinesCount Property

Gets indicator's values count

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
CountExample :
IndicatorBuilder
{
    public
CountExample()
    : base()
{
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"CountExample";
        #endregion

Lines.Set("CountExample");

base.SeparateWindow = false;
}

        BuiltInIndicator
buildIn;

        public override
void Init()
{
    buildIn =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 2,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{

Notification.Print(buildIn.Lin
es.Count.ToString());
}
}
```

See Also

Reference

BuiltInLines Class
TradeApi.Indicators Namespace

BuiltInLinesItem Property

Returns build in indicator line object by index

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInLine this[  
    int lineNumber  
] { get; }
```

[Copy](#)

Parameters

lineNumber [Int32](#)

The line number in the Lines collection

Property Value

[BuiltInLine](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
IndexerExample :  
IndicatorBuilder  
{  
    public  
IndexerExample()  
        : base()  
{  
    #region  
Initialization  
  
Credentials.ProjectName =  
"IndexerExample";  
    #endregion  
  
Lines.Set("IndexerExample");  
  
base.SeparateWindow = false;  
}  
  
BuiltInIndicator  
buildIn;  
  
    public override  
void Init()  
{  
    buildIn =  
IndicatorsManager.BuildIn.MA(H  
istoryDataSeries, 2,  
MAMode.EMA);  
}  
  
    public override  
void Update(TickStatus args)  
{  
  
Notification.Print(buildIn.Lin  
es[0].GetValue().ToString());  
}
```

```
        }  
    }
```

See Also

Reference

[BuiltInLines Class](#)

[TradeApi.Indicators Namespace](#)

BuiltInLines Methods

The [BuiltInLines](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[BuiltInLines Class](#)

[TradeApi.Indicators Namespace](#)

Chart Class

Chart entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsChart](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class Chart :  
    IDisposable
```

[Copy](#)

The [Chart](#) type exposes the following members.

► Properties

	Name	Description
	BarsWidth	Bar width value, which is on the chart window
	ChartDataType	Specifies the default source on which chart is build (ByBidAsk, ByBid, ByAsk, ByTrades etc.)
	VisibleUnits	Units (bars or ticks) count, which

is visible within the chart window

[Top](#)

◀ Methods

Name	Description
  ChartPointToScreen	Computes the location of the specified chart point into screen coordinates resulting a parent gui thread invoke
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	(Overrides ObjectFinalize)
  FindWindow(BuiltInIndicator)	Provides the index of window with specified indicator, otherwise it returns -1
  FindWindow(IndicatorBuilder)	Provides the index of

window with specified indicator, otherwise it returns -1

 FindWindows	Provides the index collection of window with specified name of indicator, otherwise it returns -1
 GetAllWindows	Retrieves all chart windows
 GetChartControl	Returns the chart's control
 GetChartPoint	Provides a chart coordinate point based on price (y-axes), time (x-axes) and any window specified by number (optional param, 0 is a default)
 GetHashCode	Serves as the default hash function. (Inherited from Object)



GetTimeValue

Provides a TimeValue structure based on the given chart point on any window specified by index (optional param, 0 is a default)



GetType

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



GUIRefresh

Forces chart refresh



PointToChart

Computes the location of the specified screen point into chart coordinates resulting main GUI thread invoke



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

Reference

[TradeApi.Indicators Namespace](#)

Chart Properties

The [Chart](#) type exposes the following members.

► Properties

	Name	Description
 	BarsWidth	Bar width value, which is on the chart window
 	ChartDataType	Specifies the default source on which chart is build (ByBidAsk, ByBid, ByAsk, ByTrades etc.)
 	VisibleUnits	Units (bars or ticks) count, which is visible within the chart window

[Top](#)

► See Also

[Reference](#)

[Chart Class](#)

TradeApi.Indicators Namespace

ChartBarsWidth Property

Bar width value, which is on the chart window

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int BarsWidth { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using Runtime.Script.Charts;
using System.Drawing;

namespace TestEnv
{
    public class
VisibleUnitsExample :
IndicatorBuilder
    {
        public
VisibleUnitsExample()
```

```
        : base()
    {
        #region Initialization

        Credentials.ProjectName =
"VisibleUnitsExample";
        #endregion

        Lines.Set("VisibleUnits");

        Lines["VisibleUnits"].Style =
LineStyle.Histogram;
    }

    public override
void Init()
{
}

public override
void Update(TickStatus args)
{
}

public override
void OnPaintChart(object
sender, PaintEventArgs
args)
{
    Lines["InitExample"].Width =
ChartSource.BarsWidth;
}
}
```

See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartChartDataType Property

Specifies the default source on which chart is build (ByBidAsk, ByBid, ByAsk, ByTrades etc.)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public ChartDataType  
ChartDataType { get; }
```

[Copy](#)

Property Value

[ChartDataType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using TradeApi.History;  
using System;  
  
namespace TestEnv  
{  
    public class  
ChartDataTypeExample :  
IndicatorBuilder
```

[Copy](#)

```
        {
            public
ChartDataTypeExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ChartDataTypeExample";
        #endregion
    }

        public override
void Init()
{
    if(ChartSource.ChartDataType
== ChartDataType.ByBid)

HistoricalDataManager.Get(new
TimeHistoricalRequest(Instrume
ntsManager.Current,
DataType.Bid, Period.Day, 1),
new
Interval(DateTime.UtcNow,
DateTime.UtcNow.AddDays(-3)));

}

        public override
void Update(TickStatus args)
{
}

}
}
```

See Also

Reference

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartVisibleUnits Property

Units (bars or ticks) count, which is visible within the chart window

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int VisibleUnits { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.History;
using System;

namespace TestEnv
{
    public class
VisibleUnitsExample :
IndicatorBuilder
{ Copy
```

```
        public
VisibleUnitsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"VisibleUnitsExample";
    #endregion
}

        public override
void Init()
{
    HistoricalDataManager.Get(new
TimeHistoricalRequest(Instруме
нtsManager.Current,
DataType.Bid, Period.Day, 1),
    new
Interval(DateTime.UtcNow,
HistoryDataSeries.GetTimeUtc(C
hartSource.VisibleUnits)));
}

        public override
void Update(TickStatus args)
{
}

}
}
```

See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

Chart Methods

The [Chart](#) type exposes the following members.

► Methods

Name	Description
  ChartPointToScreen	Computes the location of the specified chart point into screen coordinates resulting a parent gui thread invoke
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	(Overrides ObjectFinalize)
  FindWindow(BuiltInIndicator)	Provides the index of window with specified indicator, otherwise it returns -1

 FindWindow(IndicatorBuilder)	Provides the index of window with specified indicator, otherwise it returns -1
 FindWindows	Provides the index collection of window with specified name of indicator, otherwise it returns -1
 GetAllWindows	Retrieves all chart windows
 GetChartControl	Returns the chart's control
 GetChartPoint	Provides a chart coordinate point based on price (y-axes), time (x-axes) and any window specified by number (optional param, 0 is a default)
 GetHashCode	Serves as the default hash function.

(Inherited
from [Object](#))

 	GetTimeValue	Provides a TimeValue structure based on the given chart point on any window specified by index (optional param, 0 is a default)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	GUIRefresh	Forces chart refresh
 	PointToChart	Computes the location of the specified screen point into chart coordinates resulting main GUI thread invoke
 	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

↳ See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartChartPointToScreen Method

Computes the location of the specified chart point into screen coordinates resulting a parent gui thread invoke

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Point  
ChartPointToScreen(  
    Point point  
)
```

[Copy](#)

Parameters

point [Point](#)
the screen point

Return Value

[Point](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;
```

[Copy](#)

```
using System.Windows.Forms;

namespace TestEnv
{
    public class
ChartPointToScreenExample : 
IndicatorBuilder
{
    public
ChartPointToScreenExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ChartPointToScreenExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    Point
cursorPos =
ChartSource.ChartPointToScreen
(Cursor.Position);
    var timeValue
=
ChartSource.GetTimeValue(curso
rPos,
ChartSource.GetAllWindows().Fi
nd(win =>
win.IsMainWindow).WindowNumber
```

```
) ;  
  
Notification.Print(timeValue.V  
alue.ToString());  
    }  
  
}  
}
```

◀ See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartFinalize Method

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
protected override void  
Finalize()
```

[Copy](#)

Implements
[ObjectFinalize](#)

► See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartFindWindow Method

Overload List

Name	Description
 FindWindow(BuiltInIndicator)	Provides the index of window with specified indicator, otherwise it returns -1
 FindWindow(IndicatorBuilder)	Provides the index of window with specified indicator, otherwise it returns -1

[Top](#)

See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartFindWindow(BuiltInIndicator) Method

Provides the index of window with specified indicator, otherwise it returns -1

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int FindWindow(  
    BuiltInIndicator  
    indicator  
)
```

[Copy](#)

Parameters

indicator [BuiltInIndicator](#)

built in indicator object to search its location window

Return Value

[Int32](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
FindWindowExample :
IndicatorBuilder
    {
        public
FindWindowExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"FindWindowExample";
            #endregion
        }

        public override void
Init()
        {
            var windowIndex =
ChartSource.FindWindow(this);
        }

        public override void
Update(TickStatus args)
        {
            }
    }
}
```

See Also

Reference

[Chart Class](#)

[FindWindow Overload](#)

[TradeApi.Indicators Namespace](#)

ChartFindWindow(IndicatorBuilder) Method

Provides the index of window with specified indicator, otherwise it returns -1

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int FindWindow(  
    IndicatorBuilder  
    indicator  
)
```

[Copy](#)

Parameters

indicator [IndicatorBuilder](#)

built in indicator object to search its
location window

Return Value

[Int32](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
FindWindowExample :
IndicatorBuilder
    {
        public
FindWindowExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"FindWindowExample";
            #endregion
        }

        public override void
Init()
        {
            var windowIndex =
ChartSource.FindWindow(this);
        }

        public override void
Update(TickStatus args)
        {
            }
    }
}
```

See Also

Reference

[Chart Class](#)

[FindWindow Overload](#)

[TradeApi.Indicators Namespace](#)

ChartFindWindows Method

Provides the index collection of window with specified name of indicator, otherwise it returns -1

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<int> FindWindows(  
    string indicatorName  
)
```

[Copy](#)

Parameters

indicatorName [String](#)

indicator name to search its location
window

Return Value

[ListInt32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;
```

[Copy](#)

```
namespace TestEnv
{
    public class FindWindowsExample : IndicatorBuilder
    {
        public FindWindowsExample()
            : base()
        {
            #region Initialization

            Credentials.ProjectName =
                "FindWindowsExample";
            #endregion
        }

        public override void Init()
        {
            var windowIndexList =
                ChartSource.FindWindows("CCI");
        }

        public override void Update(TickStatus args)
        {
            }
    }
}
```

See Also

[Reference](#)

[Chart Class](#)

TradeApi.Indicators Namespace

Chart GetAllWindows Method

Retrieves all chart windows

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Window>
GetAllWindows()
```

[Copy](#)

Return Value

[ListWindow](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
    GetAllWindowsExample :
    IndicatorBuilder
    {
        public
        GetAllWindowsExample()
            : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "GetAllWindowsExample";  
    #endregion  
}  
  
    public override  
void Init()  
{  
    var  
windowIndexList =  
ChartSource.GetAllWindows();  
}  
  
    public override  
void Update(TickStatus args)  
{  
}  
}  
}
```

See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartGetChartControl Method

Returns the chart's control

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Object  
GetChartControl()
```

[Copy](#)

Return Value

[Object](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.History;  
using System.Windows.Forms;  
using System.Drawing;  
using System;  
using Runtime.Script.Charts;  
  
namespace TestEnv  
{  
    public class  
GetChartControlExample :
```

[Copy](#)

```
IndicatorBuilder
{
    public
GetChartControlExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetChartControlExample";
        #endregion
    }

    TextBox CustomText;
    Control chrt;

    public override
void Init()
{
    CustomText =
new TextBox();

CustomText.Location = new
Point(25, 40);

    chrt =
this.ChartSource.GetChartContr
ol() as Control;

    Action add =
() =>
{
    chrt.Controls.Add(CustomText);
};

    chrt.Invoke(add);
}
```

```
        public override
void Update(TickStatus args)
{
    CustomText.Text =
this.HistoryDataSeries.GetValu
e(PriceType.Close).ToString();
}

        public override
void OnPaintChart(object
sender, PaintEventArgs
args)
{
    Action remove
= () =>
{
    chrt.Controls.Remove(CustomTex
t);
};

chrt.Invoke(remove);
}
}
```

See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartGetChartPoint Method

Provides a chart coordinate point based on price (y-axes), time (x-axes) and any window specified by number (optional param, 0 is a default)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public PointF GetChartPoint(  
    TimeValue timeValue,  
    int windowNumber  
)  
Copy
```

Parameters

timeValue [TimeValue](#)

the cross of the time in UTC and unit value

windowNumber [Int32](#)

index of the window

Return Value

[PointF](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using System.Windows.Forms;

namespace TestEnv
{
    public class
GetChartPointExample :
IndicatorBuilder
{
    public
GetChartPointExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetChartPointExample";
        #endregion
    }

    public override
void Init()
    {

    }

    public override
void Update(TickStatus args)
    {
        Point
cursorPos =
ChartSource.PointToChart(Curso
r.Position);
        var timeValue
=
ChartSource.GetTimeValue(curso
rPos,
```

```
ChartSource.GetAllWindows().Find(win =>
    win.IsMainWindow).WindowNumber
);

Notification.Print(timeValue.Value.ToString());
}
}
}
```

See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartGetTimeValue Method

Provides a TimeValue structure based on the given chart point on any window specified by index (optional param, 0 is a default)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TimeValue GetTimeValue(Copy
    PointF pointF,
    int windowNumber
)
```

Parameters

pointF [PointF](#)

the chart point

windowNumber [Int32](#)

index of the window

Return Value

[TimeValue](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;
using Runtime.Script.Charts;
using System.Drawing;

namespace TestEnv
{
    public class
GetTimeValueExample :
IndicatorBuilder
{
    public
GetTimeValueExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetTimeValueExample";
        #endregion
    }

    public override void
Init()
{
}

    public override void
Update(TickStatus args)
{
}

    public override void
OnPaintChart(object sender,
PaintEventArgs args)
{
    var mainWidow =

```

```
ChartSource.GetAllWindows().Find(w  
in => win.IsMainWindow);  
  
        var  
windowRectUpperPoint =  
mainWidow.WindowRectangle.Location  
;  
        var  
windowHighestPriceEdge =  
ChartSource.PointToChart(windowRec  
tUpperPoint);  
        var  
timeValueHighest =  
ChartSource.GetTimeValue(windowHig  
hestPriceEdge,  
mainWidow.WindowNumber);  
        PointF  
dividerLineStart =  
ChartSource.GetChartPoint(timeValu  
eHighest, mainWidow.WindowNumber);  
  
        var  
windowLowestPriceEdge =  
ChartSource.PointToChart(new  
Point(mainWidow.WindowRectangle.Le  
ft,  
mainWidow.WindowRectangle.Bottom))  
;  
        var  
timeValueLowest =  
ChartSource.GetTimeValue(windowLow  
estPriceEdge,  
mainWidow.WindowNumber);  
        PointF  
dividerLineEnd =  
ChartSource.GetChartPoint(timeValu  
eLowest, mainWidow.WindowNumber);  
  
args.Graphics.DrawLine(new
```

```
    Pen(Color.Orange, 2),  
    dividerLineStart, dividerLineEnd);  
  
    ChartSource.GUIRefresh();  
}  
}  
}
```

See Also

Reference

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartGUIRefresh Method

Forces chart refresh

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void GUIRefresh()
```

[Copy](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using Runtime.Script.Charts;
using System.Drawing;

namespace TestEnv
{
    public class
    GUIRefreshExample :
    IndicatorBuilder
    {
        public
        GUIRefreshExample()
        : base()
        {
            #region
```

[Copy](#)

```
Initialization

Credentials.ProjectName =
"GUIRefreshExample";
#endregion
}

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
}

    public override
void OnPaintChart(object
sender, PaintEventArgs
args)
{
    var mainWidow
=
ChartSource.GetAllWindows().Fi
nd(win => win.IsMainWindow);

    var
windowRectUpperPoint =
mainWidow.WindowRectangle.Loca
tion;
    var
windowHighestPriceEdge =
ChartSource.PointToChart(windo
wRectUpperPoint);
    var
timeValueHighest =
```

```
    ChartSource.GetTimeValue(windo
wHighestPriceEdge,
mainWidow.WindowNumber) ;
                PointF
dividerLineStart =
ChartSource.GetChartPoint(time
ValueHighest,
mainWidow.WindowNumber) ;

            var
windowLowestPriceEdge =
ChartSource.PointToChart(new
Point(mainWidow.WindowRectangl
e.Left,
mainWidow.WindowRectangle.Bott
om));
            var
timeValueLowest =
ChartSource.GetTimeValue(windo
wLowestPriceEdge,
mainWidow.WindowNumber) ;
                PointF
dividerLineEnd =
ChartSource.GetChartPoint(time
ValueLowest,
mainWidow.WindowNumber) ;

args.Graphics.DrawLine(new
Pen(Color.Orange, 2),
dividerLineStart,
dividerLineEnd);

ChartSource.GUIRefresh();
        }
    }
}
```

See Also

Reference

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartPointToChart Method

Computes the location of the specified screen point into chart coordinates resulting main GUI thread invoke

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Point PointToChart(Copy
    Point point
)
```

Parameters

point [Point](#)
the chart point

Return Value
[Point](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Indicators;
using System.Drawing;
using System.Windows.Forms;
```

```
namespace TestEnv
{
    public class
PointToChartExample : 
IndicatorBuilder
{
    public
PointToChartExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"PointToChartExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    Point
cursorPos =
ChartSource.PointToChart(Curso
r.Position);
    var timeValue
=
ChartSource.GetTimeValue(curso
rPos,
ChartSource.GetAllWindows().Fi
nd(win =>
win.IsMainWindow).WindowNumber
);
}
```

```
Notification.Print(timeValue.V  
alue.ToString());  
}  
}  
}
```

See Also

[Reference](#)

[Chart Class](#)

[TradeApi.Indicators Namespace](#)

ChartDataType Enumeration

Chart data source type

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum ChartDataType
```

[Copy](#)

► Members

Member name	Value	Description
ByBid	1	ByBid
ByTrades	3	ByTrades
ByAsk	4	ByAsk

► See Also

Reference

[TradeApi.Indicators Namespace](#)

Cloud Class

Cloud entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsCloud](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class Cloud
```

[Copy](#)

The `Cloud` type exposes the following members.

▪ Properties

	Name	Description
	ColorAscending	Cloud's ascending color
	ColorDescending	Cloud's descending color
	GradientMode	Cloud's gradient mode

 	TimeShift	Cloud's time shift
---	---------------------------	--------------------

 	ValueShift	Cloud's value shift
---	----------------------------	---------------------

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
 	Set	Sets cloud
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

Cloud Properties

The [Cloud](#) type exposes the following members.

► Properties

	Name	Description
 	ColorAscending	Cloud's ascending color
 	ColorDescending	Cloud's descending color
 	GradientMode	Cloud's gradient mode
 	TimeShift	Cloud's time shift
 	ValueShift	Cloud's value shift

[Top](#)

► See Also

[Reference](#)

[Cloud Class](#)

[TradeApi.Indicators Namespace](#)

CloudColorAscending Property

Cloud's ascending color

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color ColorAscending {  
    get; set; }
```

[Copy](#)

Property Value

Color

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{  
    public class  
ColorAscendingExample :  
IndicatorBuilder  
    {  
        public  
ColorAscendingExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ColorAscendingExample";
        #endregion

        Clouds.Set("SetExample",
        GradientMode.Linear,
        Color.Red, Color.Orange);

        base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{

Clouds["CloudsExample"].ColorA
scending = Color.Blue;
}
}
}
```

See Also

[Reference](#)

[Cloud Class](#)

[TradeApi.Indicators Namespace](#)

CloudColorDescending Property

Cloud's descending color

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color ColorDescending {Copy
    get; set; }
```

Property Value

Color

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
ColorDescendingExample : IndicatorBuilder
    {
        public
ColorDescendingExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ColorDescendingExample";
        #endregion

        Clouds.Set("SetExample",
        GradientMode.Linear,
        Color.Red, Color.Orange);

        base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{

Clouds["CloudsExample"].ColorD
escending = Color.Green;
}
}

}
```

See Also

[Reference](#)

[Cloud Class](#)

[TradeApi.Indicators Namespace](#)

CloudGradientMode Property

Cloud's gradient mode

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public GradientMode
GradientMode { get; set; }
```

[Copy](#)

Property Value

[GradientMode](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
GradientModeExample :
IndicatorBuilder
{
    public
GradientModeExample()
```

[Copy](#)

```
        : base()
    {
        #region Initialization

        Credentials.ProjectName =
"GradientModeExample";
        #endregion

        Clouds.Set("TimeShift",
GradientMode.Linear,
Color.Red, Color.Orange);

        base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{

Clouds["CloudsExample"].GradientMode =
GradientMode.Simple;
}
}
}
```

See Also

[Reference](#)

[Cloud Class](#)

[TradeApi.Indicators Namespace](#)

CloudTimeShift Property

Cloud's time shift

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int TimeShift { get;  
    set; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{  
    public class  
TimeShiftExample :  
IndicatorBuilder  
    {  
        public  
TimeShiftExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "TimeShiftExample";
        #endregion

        Clouds.Set("TimeShift",
        GradientMode.Linear,
        Color.Red, Color.Orange);

        base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{

Clouds["CloudsExample"].TimeSh
ift = 5;
}
}
}
```

See Also

[Reference](#)

[Cloud Class](#)

[TradeApi.Indicators Namespace](#)

CloudValueShift Property

Cloud's value shift

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double ValueShift {  
    get; set; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{  
    public class  
ValueShiftExample :  
IndicatorBuilder  
    {  
        public  
ValueShiftExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ValueShiftExample";
        #endregion

        Clouds.Set("ValueShiftExample"
        , GradientMode.Linear,
        Color.Red, Color.Orange);

        base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{

Clouds["CloudsExample"].ValueShift =
InstrumentsManager.Current.Day
Info.High;
}
}
}
```

See Also

Reference

Cloud Class
TradeApi.Indicators Namespace

Cloud Methods

The [Cloud](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	Set	Sets cloud
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[Cloud Class](#)

[TradeApi.Indicators Namespace](#)

CloudSet Method

Sets cloud

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void Set(Copy
    double value1,
    double value2,
    int offset = 0
)
```

Parameters

value1 Double

value2 Double

offset Int32 (Optional)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
```

```
{  
    public class SetExample  
: IndicatorBuilder  
    {  
        public SetExample()  
        : base()  
        {  
            #region  
Initialization  
  
Credentials.ProjectName =  
"SetExample";  
            #endregion  
  
Clouds.Set("SetExample",  
GradientMode.Linear,  
Color.Red, Color.Orange);  
  
base.SeparateWindow = false;  
    }  
  
        public override  
void Init()  
    {  
  
    }  
  
        public override  
void Update(TickStatus args)  
    {  
  
Clouds["CloudsExample"].Set(  
InstrumentsManager.Current.Day  
Info.High,  
InstrumentsManager.Current.Day  
Info.Low);  
    }
```

```
        }  
    }
```

See Also

[Reference](#)

[Cloud Class](#)

[TradeApi.Indicators Namespace](#)

Clouds Class

Clouds entity

▪ Inheritance Hierarchy

[SystemObject](#) TradeApi.IndicatorsCloud
s

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class Clouds
```

[Copy](#)

The [Clouds](#) type exposes the following members.

▪ Properties

	Name	Description
	Count	Clouds count
	ItemInt32	Clouds indexer
	ItemString	Clouds indexer

[Top](#)

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	Set	Cloud set
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

Clouds Properties

The [Clouds](#) type exposes the following members.

Properties

	Name	Description
 	Count	Clouds count
 	ItemInt32	Clouds indexer
 	ItemString	Clouds indexer

[Top](#)

See Also

[Reference](#)

[Clouds Class](#)

[TradeApi.Indicators Namespace](#)

CloudsCount Property

Clouds count

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

Int32

◀ Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
CountExample :
IndicatorBuilder
{
    public
CountExample()
    : base()
{
    #region
```

[Copy](#)

```
Initialization

Credentials.ProjectName =
"CountExample";
#endregion

Clouds.Set("CloudsExample",
GradientMode.Linear,
Color.Red, Color.Orange);

base.SeparateWindow = false;
}

public override
void Init()
{
    Notification.Print(Clouds.Count.ToString());
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[Clouds Class](#)

[TradeApi.Indicators Namespace](#)

CloudsItem Property

▪ Overload List

Name	Description
 ItemInt32	Clouds indexer
 ItemString	Clouds indexer

[Top](#)

▪ See Also

[Reference](#)

[Clouds Class](#)

[TradeApi.Indicators Namespace](#)

CloudsItem(Int32) Property

Clouds indexer

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Cloud this[  
    int index  
] { get; }
```

[Copy](#)

Parameters

index [Int32](#)
cloud index

Property Value
[Cloud](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
CloudExample :
IndicatorBuilder
{
    public
CloudExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"CloudExample";
        #endregion

Clouds.Set("CloudsExample",
GradientMode.Linear,
Color.Red, Color.Orange);

base.SeparateWindow = false;
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
}

Clouds[0].ColorAscending =
Color.Green;
}
}
```

See Also

[Reference](#)

[Clouds Class](#)

[Item Overload](#)

[TradeApi.Indicators Namespace](#)

CloudsItem(String) Property

Clouds indexer

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Cloud this[  
    string name  
] { get; }
```

[Copy](#)

Parameters

name [String](#)
cloud name

Property Value

[Cloud](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
CloudExample :
IndicatorBuilder
{
    public
CloudExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"CloudExample";
        #endregion

Clouds.Set("CloudsExample",
GradientMode.Linear,
Color.Red, Color.Orange);

base.SeparateWindow = false;
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    Clouds["CloudsExample"].ColorA
scending = Color.Green;
}
}
```

See Also

[Reference](#)

[Clouds Class](#)

[Item Overload](#)

[TradeApi.Indicators Namespace](#)

Clouds Methods

The [Clouds](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	Set	Cloud set
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[Clouds Class](#)

[TradeApi.Indicators Namespace](#)

CloudsSet Method

Cloud set

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public void Set(Copy
                string name,
                GradientMode
gradientMode,
                Color colorAscending,
                Color colorDescending
            )
```

Parameters

name String

 cloud name

gradientMode GradientMode

 cloud gradient

colorAscending Color

 gradient ascending color

colorDescending Color

 gradient descending color

◀ Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class SetExample : IndicatorBuilder
    {
        public SetExample()
        : base()
        {
            #region Initialization
            Credentials.ProjectName =
"SetExample";
            #endregion

            Clouds.Set("CloudsExample",
GradientMode.Linear, Color.Red,
Color.Orange);

            base.SeparateWindow
= false;
        }

        public override void
Init()
{
}

        public override void
Update(TickStatus args)
{
}
}
```

```
        }  
    }
```

See Also

[Reference](#)

[Clouds Class](#)

[TradeApi.Indicators Namespace](#)

GradientMode Enumeration

Cloud's gradient mode

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum GradientMode
```

[Copy](#)

► Members

Member name	Value	Description
None	0	None
Simple	1	Simple
Linear	2	Linear

► See Also

Reference

[TradeApi.Indicators Namespace](#)

IBarStampElement Interface

IBarStampElement interface

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public interface  
IBarStampElement
```

[Copy](#)

The [IBarStampElement](#) type exposes the following members.

► Properties

	Name	Description
 Type	Type	Type of the barstamp

[Top](#)

► See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

IBarStampElement Properties

The [IBarStampElement](#) type exposes the following members.

Properties

	Name	Description
 Type	Type	Type of the barstamp

[Top](#)

See Also

[Reference](#)

[IBarStampElement Interface](#)

[TradeApi.Indicators Namespace](#)

IBarStamp ElementType Property

Type of the barstamp

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
BarStampType Type { get; }
```

[Copy](#)

Property Value

[BarStampType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.History;
using System.Drawing;

namespace TestEnv
{
    public class
BarColorExample :
IndicatorBuilder
{
    public
BarColorExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "BarColorExample";
        #endregion

        base.SeparateWindow = false;
    }

        BuiltInIndicator
ema;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{
    var barStamp =
new BarStampColor(Color.Gray,
Color.Gray, 0);
    if
(barStamp.Type ==
BarStampType.BarColor &&
ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampColor(Color.Green,
```

```
Color.Green, 0));
        else if
    (barStamp.Type ==
BarStampType.BarColor &&
ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampColor(Color.Red,
Color.Red, 0));
}
}
}
```

See Also

Reference

[IBarStampElement Interface](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilder Class

Provides access to the indicator's kit of the platform

▪ Inheritance Hierarchy

```
SystemObject  ScriptBase
CSharpScript
CSharpIndicator
TradeApi.ToolBeltBaseScript
TradeApi.IndicatorsIndicatorBuilder
```

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public class IndicatorBuilder : BaseScript
```

[Copy](#)

The `IndicatorBuilder` type exposes the following members.

▪ Constructors

	Name	Description
	IndicatorBuilder	Constructor

[Top](#)

▪ Properties

	Name	Description
	AccountManager	A manager reference to retrieve accounts (Inherited from <code>BaseScript</code>)
	BarStamps	Gets a collection of bar stamps
	ChartSource	Represents access to the current chart
	Clouds	Gets a collection of clouds
	Connection	Connection entity (Inherited from <code>BaseScript</code>)
	Credentials	Script credentials (Inherited from <code>CSharpScript</code>)
	Disposed	(Inherited from <code>CSharpScript</code>)
	HistoricalDataManager	A historicaldata manager reference

		to wield custom control over extra required historical element (Inherited from BaseScript)
	HistoryDataSeries	Represent access to instrument's historical data based on current chart (Inherited from BaseScript)
	Id	(Inherited from CSharpScript)
	IndicatorsManager	An indicator manager reference to obtain usage of buildin and custom indicators (Inherited from BaseScript)
	InstrumentsManager	A manager reference to query and manage vendor's instruments (Inherited from BaseScript)
	Lines	Gets a collection of an indicator lines
	Notification	An indicator's notification hub (Inherited from BaseScript)
	OrdersViewer	Provides viewer for all orders
	PositionsViewer	Provides viewer for all positions
	ProjectType	(Inherited from CSharpScript)
	ScriptShortName	Short name of the script
	SeparateWindow	(Inherited from CSharpIndicator)
	SingleThreadEvents	(Inherited from CSharpScript)
	Statistic	Statistic entity (Inherited from BaseScript)
	Thresholds	Gets a collection of thresholds lines
	Tools	Tool assistance in platform (Inherited from BaseScript)

[Top](#)

Methods

Name	Description
 Complete	This function is called before r script (Inherited from BaseScript)

Dispose

(Inherited from **CSharpScript**)

≡♪ Equals	Determines whether the specified value is equal to the current object. (Inherited from Object)
≡♪ Finalize	Allows an object to try to free memory and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
≡♪ GetArray(ArrayList)	(Inherited from ScriptBase)
≡♪ GetArray(ArrayList, Boolean)	(Inherited from ScriptBase)
≡♪ GetArray(Type, Int32)	(Inherited from ScriptBase)
≡♪ GetHashCode	Serves as the default hash function. (Inherited from Object)
≡♪ GetType	Gets the Type of the current instance. (Inherited from Object)
≡♪ ≡ Init	This function is called once before the work starts. (Inherited from BaseScript)
≡♪ LoadAndInvoke	(Inherited from CSharpScript)
≡♪ MemberwiseClone	Creates a shallow copy of the Object . (Inherited from Object)
≡♪ OnDebug	(Inherited from CSharpScript)
≡♪ OnPaintChart	(Inherited from CSharpScript)
≡♪ PushMethodDump	(Inherited from CSharpScript)
≡♪ QueueUserWorkItem(WaitCallback)	(Inherited from CSharpScript)
≡♪ QueueUserWorkItem(WaitCallback, Object)	(Inherited from CSharpScript)
≡♪ QueueUserWorkItem(WaitCallback, Object, ApartmentState)	(Inherited from CSharpScript)
≡♪ RegisterLocalVariable(String, FuncObject, String)	(Inherited from CSharpScript)

 RegisterLocalVariable(String, FuncObject, ActionObject, String)	(Inherited from CSharpScript)
 SetIndicatorsDigits	Sets count of digits after decimal point to visualize indicator values
 ToString	Returns a string that represents the current object. (Inherited from Object)
 Update	This function is called amid new bar or history processing. It takes argument with a tick status to quote state (args.TickStatus, IsHistory/IsBar). (Inherited from BaseScript)

[Top](#)

Fields

Name	Description
 currentInternalInstrument	(Inherited from BaseScript)
 ScriptCache	(Inherited from CSharpScript)

[Top](#)

See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilder Constructor

Constructor

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public IndicatorBuilder()
```

[Copy](#)

► See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilder Properties

The [IndicatorBuilder](#) type exposes the following members.

Properties

Name	Description
  AccountManager	A manager reference to retrieve accounts (Inherited from BaseScript)
  BarStamps	Gets a collection of an bar stamps
  ChartSource	Represents access to the current chart
  Clouds	Gets a collection of clouds
 Connection	Connection entity (Inherited from BaseScript)
 Credentials	Script credentials (Inherited from CSharpScript)
 Disposed	(Inherited from

CSharpScript)

	HistoricalDataManager	A historical data manager reference to wield custom control over extra required historical element (Inherited from BaseScript)
	HistoryDataSeries	Represent access to instrument's historical data based on current chart (Inherited from BaseScript)
	Id	(Inherited from CSharpScript)
	IndicatorsManager	An indicator manager reference to obtain usage of buildin and custom indicators (Inherited from BaseScript)
	InstrumentsManager	A manager reference to query and manage vendor's instruments (Inherited from BaseScript)
	Lines	Gets a collection of

an indicator lines

 Notification	An indicator's notification hub (Inherited from BaseScript)
 OrdersViewer	Provides viewer for all orders
 PositionsViewer	Provides viewer for all positions
 ProjectType	(Inherited from CSharpScript)
 ScriptShortName	Short name of the script
 SeparateWindow	(Inherited from CSharpIndicator)
 SingleThreadEvents	(Inherited from CSharpScript)
 Statistic	Statistic entity (Inherited from BaseScript)
 Thresholds	Gets a collection of thresholds lines
 Tools	Tool assistance in platform (Inherited from BaseScript)

[Top](#)

◀ See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilderBar Stamps Property

Gets a collection of an bar stamps

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BarStamps BarStamps {  
    get; }
```

[Copy](#)

Property Value

[BarStamps](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.History;  
using System.Drawing;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
BarStampsExample :  
IndicatorBuilder  
    {  
        public
```

```
BarStampsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BarStampsExample";
    #endregion

Lines.Set("BarStampsExample");

base.SeparateWindow = false;
}

        BuiltInIndicator
ema;

        public override
void Init()
{
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{
    if
(ema.GetValue() >
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampColor(Color.Green,
Color.Green, 0));
}
```

```
        else if
(ema.GetValue() <
HistoryDataSeries.GetValue(Pri
ceType.Close, 1))

BarStamps.Set(new
BarStampColor(Color.Red,
Color.Red, 0));
}
}
```

See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilderChart Source Property

Represents access to the current chart

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Chart ChartSource {  
    get; }
```

[Copy](#)

Property Value
[Chart](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Windows.Forms;  
using System;  
using System.Drawing;  
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
ChartSourceExample :  
IndicatorBuilder
```

```
        {
            public
ChartSourceExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ChartSourceExample";
        #endregion

Lines.Set("ChartSourceExample"
);

base.SeparateWindow = false;
    }

    TextBox CustomText;
    Control chrt;

        public override
void Init()
{
    CustomText =
new TextBox();

CustomText.Location = new
Point(25, 40);

    chrt =
this.ChartSource.GetChartContr
ol() as Control;

        Action add =
() =>
{

```

```
        chrt.Controls.Add(CustomText);
    }

    chrt.Invoke(add);
}

public override
void Update(TickStatus args)
{
    CustomText.Text =
this.HistoryDataSeries.GetValu
e(PriceType.Close).ToString();
}

public override
void Complete()
{
    Action remove
= () =>
{
    chrt.Controls.Remove(CustomTex
t);
}
}

chrt.Invoke(remove);
}
}
```

See Also

Reference

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilderClouds Property

Gets a collection of clouds

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Clouds Clouds { get; }
```

[Copy](#)

Property Value

[Clouds](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;
using TradeApi;
using TradeApi.History;

namespace TestEnv
{
    public class
CloudsExample :
IndicatorBuilder
{
    public
```

[Copy](#)

```
CloudsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"CloudsExample";
    #endregion

Clouds.Set("CloudsExample",
GradientMethod,
Color.OrangeRed,
Color.CadetBlue);

base.SeparateWindow = false;
}

[InputParameter(InputType.Combobox, "Gradient method", 3)]

[ComboboxItem("Simple",
GradientMode.Simple)]

[ComboboxItem("Linear",
GradientMode.Linear)]

[ComboboxItem("None",
GradientMode.None)]
    public GradientMode
GradientMethod =
GradientMode.Simple;

    public override
void Init()
{
}

}
```

```
        public override
void Update(TickStatus args)
{
    var high =
HistoryDataSeries.GetValue(Pri
ceType.High);
    var low =
HistoryDataSeries.GetValue(Pri
ceType.Low);

Clouds["CloudsExample"].Set(hi
gh, low);
}
}
```

See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilderLines Property

Gets a collection of an indicator lines

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Lines Lines { get; }
```

[Copy](#)

Property Value

[Lines](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System;

namespace TestEnv
{
    public class
LinesExample :
IndicatorBuilder
{
    public
LinesExample()
    : base()
```

[Copy](#)

```
{  
    #region  
Initialization  
  
    Credentials.ProjectName =  
    "LinesExample";  
    #endregion  
  
    Lines.Set("LinesExample");  
  
    base.SeparateWindow = false;  
}  
  
Random rand;  
  
public override  
void Init()  
{  
    rand = new  
Random();  
}  
  
public override  
void Update(TickStatus args)  
{  
  
    this.Lines["LinesExample"].Set  
Value(rand.Next()); // sets of  
the element in Lines  
collection  
  
    Notification.Print(this.Lines.  
Count.ToString()); // gets  
total lines count  
}  
}  
}
```

See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilderOrders Viewer Property

Provides viewer for all orders

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public OrdersViewer  
OrdersViewer { get; }
```

[Copy](#)

Property Value

[OrdersViewer](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OrdersViewerExample :  
IndicatorBuilder  
    {  
        public  
OrdersViewerExample()  
        : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "OrdersViewerExample";
            #endregion
        }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

        Notification.Print($"Total
sell:
{OrdersViewer.TotalSellQuantit
y}");

        Notification.Print($"Total
buy:
{OrdersViewer.TotalBuyQuantity
} ");
    }
}
}
```

See Also

Reference

IndicatorBuilder Class
TradeApi.Indicators Namespace

IndicatorBuilderPositionsViewer Property

Provides viewer for all positions

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public PositionsViewer  
    PositionsViewer { get; }
```

[Copy](#)

Property Value

[PositionsViewer](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
    PositionsViewerExample :  
    IndicatorBuilder  
    {  
        public
```

[Copy](#)

```
    PositionsViewerExample()
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"PositionsViewerExample";
        #endregion
    }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{

    Notification.Print($"Total
sell position:
{PositionsViewer.GetPositions(
position => position.Side ==
PositionSide.Short)}");

    Notification.Print($"Total buy
position:
{PositionsViewer.GetPositions(
position => position.Side ==
PositionSide.Long)}");
}
}
}
```

See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilderScript ShortName Property

Short name of the script

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string ScriptShortName
{ get; set; }
```

[Copy](#)

Property Value

String

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
ScriptShortNameExample :
IndicatorBuilder
{
    public
ScriptShortNameExample()
    : base()
```

[Copy](#)

```
{  
    #region  
Initialization  
  
Credentials.ProjectName =  
"ScriptShortNameExample";  
    #endregion  
  
Lines.Set("ScriptShortNameExam  
ple");  
  
base.SeparateWindow = false;  
}  
  
[InputParameter(InputType.Nume  
ric, "Period", 0)]  
    [SimpleNumeric(1D,  
99999D)]  
        public int Period =  
13;  
  
        public override  
void Init()  
{  
  
ScriptShortName = $"Shorty:  
{Period}";  
}  
  
        public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

Indicator BuilderThresholds Property

Gets a collection of thresholds lines

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Thresholds Thresholds {  
    get; }
```

[Copy](#)

Property Value

[Thresholds](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
ThresholdsExample :  
IndicatorBuilder  
    {  
        public
```

[Copy](#)

```
ThresholdsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ThresholdsExample";
    #endregion

Thresholds.Set("Upper bound");
Thresholds.Set("Lower bound");
Thresholds["Upper
bound"].Level = 80;
Thresholds["Lower
bound"].Level = 20;

base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilder Methods

The [IndicatorBuilder](#) type exposes the following members.

Methods

Name	Description
 Complete	This function is called before the script starts. (Inherited from BaseScript)
 Dispose	(Inherited from CSharpScript)
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	Allows an object to try to free up memory and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
 GetArray(ArrayList)	(Inherited from ScriptBase)
 GetArray(ArrayList, Boolean)	(Inherited from ScriptBase)
 GetArray(Type, Int32)	(Inherited from ScriptBase)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 Init	This function is called once before the script starts. (Inherited from BaseScript)
 LoadAndInvoke	(Inherited from CSharpScript)
 MemberwiseClone	Creates a shallow copy of the current instance. (Inherited from Object)
 OnDebug	(Inherited from CSharpScript)
 OnPaintChart	(Inherited from CSharpScript)

PushMethodDump	(Inherited from CSharpScript)
QueueUserWorkItem(WaitCallback)	(Inherited from CSharpScript)
QueueUserWorkItem(WaitCallback, Object)	(Inherited from CSharpScript)
QueueUserWorkItem(WaitCallback, Object, ApartmentState)	(Inherited from CSharpScript)
RegisterLocalVariable(String, FuncObject, String)	(Inherited from CSharpScript)
RegisterLocalVariable(String, FuncObject, ActionObject, String)	(Inherited from CSharpScript)
SetIndicatorsDigits	Sets count of digits after decimal point to visualize indicator values
ToString	Returns a string that represents the current object. (Inherited from Object)
Update	This function is called amid new bar or history processing. It takes one argument with a tick status to quote state (args.TickStatus, IsHistory/IsBar). (Inherited from BaseScript)

[Top](#)

See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorBuilderSet IndicatorsDigits Method

Sets count of digits after decimal point to visualize indicator values

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void
SetIndicatorsDigits(
    int digits
)
```

[Copy](#)

Parameters

digits [Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
```

[Copy](#)

```
SetIndicatorsDigitsExample :  
IndicatorBuilder  
{  
    public  
SetIndicatorsDigitsExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"SetIndicatorsDigitsExample";  
        #endregion  
  
Lines.Set("SetIndicatorsDigits  
Example");  
  
base.SeparateWindow = false;  
    }  
  
    public override  
void Init()  
    {  
  
SetIndicatorsDigits(2);  
    }  
  
    public override  
void Update(TickStatus args)  
    {  
  
    }  
}
```

See Also

Reference

IndicatorBuilder Class
TradeApi.Indicators Namespace

IndicatorBuilder Fields

The [IndicatorBuilder](#) type exposes the following members.

↳ Fields

	Name	Description
💡	currentInternalInstrument	(Inherited from BaseScript)
💡	ScriptCache	(Inherited from CSharpScript)

[Top](#)

↳ See Also

[Reference](#)

[IndicatorBuilder Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLine Class

IndicatorLine entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsIndicatorLine](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
IndicatorLine
```

[Copy](#)

The [IndicatorLine](#) type exposes the following members.

▪ Properties

	Name	Description
	ArrowCode	Sets the code for the symbol type line
	Color	Line color
	Item	Lines indexer
	Markers	Sets the marker

of the line

	Name	Line name
	Style	Line style
	TimeShift	Line time shift
	ValueShift	Line value shift
	Visible	Line visibility
	Width	Sets the width of the line

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance.

(Inherited
from [Object](#))

 	GetValue	Gets line value
 	SetEmptyValue	Sets the unset value to the line series
 	SetLineBegin	Sets the starting point to draw the line
 	SetValue	Sets line value
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

IndicatorLine Properties

The [IndicatorLine](#) type exposes the following members.

Properties

	Name	Description
 	ArrowCode	Sets the code for the symbol type line
 	Color	Line color
 	Item	Lines indexer
 	Markers	Sets the marker of the line
 	Name	Line name
 	Style	Line style
 	TimeShift	Line time shift
 	ValueShift	Line value shift
 	Visible	Line visibility
 	Width	Sets the width of the line

[Top](#)

See Also

[Reference](#)

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineArrow Code Property

Sets the code for the symbol type line

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int ArrowCode { get;  
    set; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
ArrowCodeExample :  
IndicatorBuilder  
    {  
        public  
ArrowCodeExample()  
            : base()
```

[Copy](#)

```
{  
    #region  
Initialization  
  
    Credentials.ProjectName =  
    "ArrowCodeExample";  
    #endregion  
  
    Lines.Set("ArrowCodeExample");  
  
    Lines["ArrowCodeExample"].Style = LineStyle.Symbol;  
  
    Lines["ArrowCodeExample"].ArrowCode = 041809;  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
    Lines["ArrowCodeExample"].Width = 2;  
}  
  
public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

[Reference](#)

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineColor Property

Line color

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color Color { get; set; } Copy
```

Property Value

Color

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
ColorExample :
IndicatorBuilder
{
    public
ColorExample()
```

```
: base()
{
    #region
Initialization

Credentials.ProjectName =
"ColorExample";
    #endregion

Lines.Set("LinesExample");

base.SeparateWindow = false;
}

public override
void Init()
{
    Lines["LinesExample"].Color =
Color.Orange;
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineItem Property

Lines indexer

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double this[  
    int offset  
] { get; set; }
```

[Copy](#)

Parameters

offset [Int32](#)
unit index offset

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
LinesExample :  
IndicatorBuilder  
{  
    public  
LinesExample()  
        : base()  
{  
            #region  
Initialization  
  
Credentials.ProjectName =  
"LinesExample";  
            #endregion  
  
Lines.Set("VisibleExample");  
  
base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
}  
  
public override  
void Update(TickStatus args)  
{  
  
Lines["LinesExample"][0] =  
InstrumentsManager.Current.Day  
Info.Ask;  
    var lineValue  
= Lines["LinesExample"][0];  
    }  
}  
}
```

See Also

[Reference](#)

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineMarkers Property

Sets the marker of the line

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Markers Markers { get;  
}
```

[Copy](#)

Property Value
[Markers](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{  
    public class  
    MarkersExample :  
    IndicatorBuilder  
    {  
        public  
        MarkersExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "MarkersExample";
        #endregion

        Lines.Set("MarkersExample");

        base.SeparateWindow = false;
    }

        public override
void Init()
{
}

Lines["MarkersExample"].Markers.Set(Color.Orange, 0);
}

        public override
void Update(TickStatus args)
{
}

}
}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineName Property

Line name

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Name { get; }
```

[Copy](#)

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
NameExample : IndicatorBuilder
    {
        public
NameExample()
            : base()
        {
            #region
```

[Copy](#)

```
Initialization

Credentials.ProjectName =
"NameExample";
#endregion

Lines.Set("LinesExample");

base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    Notification.Print(Lines["Line
sExample"].Name.ToString());
}
}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineStyle Property

Line style

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public LineStyle Style { get;  
    set; }
```

[Copy](#)

Property Value

[LineStyle](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
StyleExample :  
IndicatorBuilder  
    {  
        public  
StyleExample()  
        : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "StyleExample";
            #endregion

            Lines.Set("LinesExample");

            base.SeparateWindow = false;
        }

        public override
void Init()
{
    Lines["LinesExample"].Style =
LineStyle.Dot;
}

        public override
void Update(TickStatus args)
{
}

}

}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineTimeShift Property

Line time shift

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int TimeShift { get;  
    set; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
TimeShiftExample :  
IndicatorBuilder  
    {  
        public  
TimeShiftExample()  
            : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "TimeShiftExample";  
    #endregion  
  
    Lines.Set("LinesExample");  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
    Lines["LinesExample"].TimeShift = 2;  
}  
  
public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineValue Shift Property

Line value shift

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double ValueShift {  
    get; set; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
ValueShiftExample :  
IndicatorBuilder  
    {  
        public  
ValueShiftExample()  
            : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "ValueShiftExample";  
    #endregion  
  
    Lines.Set("LinesExample");  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
    Lines["LinesExample"].ValueShift =  
    InstrumentsManager.Current.MinimumTickSize*2;  
}  
  
public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineVisible Property

Line visibility

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool Visible { get;  
    set; }
```

[Copy](#)

Property Value

[Boolean](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
VisibleExample :  
IndicatorBuilder  
    {  
        public  
VisibleExample()  
            : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "VisibleExample";  
    #endregion  
  
    Lines.Set("LinesExample");  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
    Lines["LinesExample"].Visible  
    = true;  
}  
  
public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineWidth Property

Sets the width of the line

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Width { get; set; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
WidthExample :
IndicatorBuilder
{
    public
WidthExample()
    : base()
{
```

```
        #region
Initialization

Credentials.ProjectName =
"WidthExample";
        #endregion

Lines.Set("WidthExample");

base.SeparateWindow = false;
}

    public override
void Init()
{
    Lines["WidthExample"].Width =
2;
}

    public override
void Update(TickStatus args)
{
}

}

}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLine Methods

The [IndicatorLine](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	GetValue	Gets line value
	SetEmptyValue	Sets the unset value

to the line
series



[SetLineBegin](#)

Sets the
starting point
to draw the
line



[SetValue](#)

Sets line
value



[ToString](#)

Returns a
string that
represents
the current
object.
(Inherited
from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineGetValue Method

Gets line value

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GetValue(Copy
                      int offset = 0
                    )
```

Parameters

offset [Int32](#) (Optional)
unit index offset

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Indicators;

namespace TestEnv
{
    public class
```

```
GetValueExample :  
IndicatorBuilder  
{  
    public  
    GetValueExample()  
        : base()  
    {  
        #region  
        Initialization  
  
        Credentials.ProjectName =  
        "GetValueExample";  
        #endregion  
  
        Lines.Set("LinesExample");  
  
        base.SeparateWindow = false;  
    }  
  
    public override  
    void Init()  
    {  
    }  
  
    public override  
    void Update(TickStatus args)  
    {  
        var lineValue  
        =  
        Lines["LinesExample"].GetValue  
        ();  
    }  
}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorLineSet EmptyValue Method

Sets the unset value to the line series

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void SetEmptyValue(Copy
                           int offset = 0
                         )
```

Parameters

offset [Int32](#) (Optional)
value to set

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;

namespace TestEnv
{
    public class
SetEmptyValueExample :  
IndicatorBuilder
    {
```

```
        public
SetEmptyValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"SetEmptyValueExample";
    #endregion

Lines.Set("SetEmptyValue");

base.SeparateWindow = false;
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{

if(HistoryDataSeries.Count <
20)

Lines["SetEmptyValue"].SetEmpt
yValue(0);
}
}
```

See Also

Reference

IndicatorLine Class
TradeApi.Indicators Namespace

IndicatorLineSetLine Begin Method

Sets the starting point to draw the line

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void SetLineBegin(Copy
    int offset = 0
)
```

Parameters

offset [Int32](#) (Optional)
value to set

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Indicators;

namespace TestEnv
{
    public class
SetLineBeginExample :
IndicatorBuilder
{
```

```
        public
SetLineBeginExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"SetLineBeginExample";
    #endregion

Lines.Set("SetLineBeginExample
");
}

base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
}

Lines["SetLineBeginExample"].S
etLineBegin(20);
}
}
```

◀ See Also

Reference

[IndicatorLine Class](#)

TradeApi.Indicators Namespace

IndicatorLineSetValue Method

Sets line value

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void SetValue(Copy
    double value,
    int offset = 0
)
```

Parameters

value [Double](#)

 value to set

offset [Int32](#) (Optional)

 unit index offset

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
```

```
SetValueExample :  
IndicatorBuilder  
{  
    public  
    SetValueExample()  
        : base()  
    {  
        #region  
        Initialization  
  
        Credentials.ProjectName =  
        "SetValueExample";  
        #endregion  
  
        Lines.Set("LinesExample");  
  
        base.SeparateWindow = false;  
    }  
  
    public override  
    void Init()  
    {  
    }  
  
    public override  
    void Update(TickStatus args)  
    {  
        Lines["LinesExample"].SetValue  
        (InstrumentsManager.Current.Da  
        yInfo.Ask);  
    }  
}
```

See Also

Reference

[IndicatorLine Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorsManager Class

Manager dedicated to maintain all of indicators

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsIndicator](#)
[sManager](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class  
IndicatorsManager
```

[Copy](#)

The [IndicatorsManager](#) type exposes the following members.

► Properties

Name	Description
 BuildIn	Returns all build in indicators

[Top](#)

► Methods

Name	Description
 CreateInstance(IndicatorBuilder, Object, HistoricalData)	Returns specific indicator by name from

		the script lookup
 	CreateInstance(String, Object, HistoricalData)	Returns specific indicator by name from script lookup
 	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

Reference

[TradeApi.Indicators Namespace](#)

IndicatorsManager Properties

The [IndicatorsManager](#) type exposes the following members.

► Properties

	Name	Description
 	BuildIn	Returns all build in indicators

[Top](#)

► See Also

[Reference](#)

[IndicatorsManager Class](#)

[TradeApi.Indicators Namespace](#)

Indicators ManagerBuildIn Property

Returns all build in indicators

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicators  
BuildIn { get; }
```

[Copy](#)

Property Value

[BuiltInIndicators](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
BuildInExample :  
IndicatorBuilder  
    {  
        public
```

[Copy](#)

```
BuildInExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BuildInExample";
    #endregion

Lines.Set("BuildInExample");

base.SeparateWindow = false;
}

        BuiltInIndicator
buildIn;

        public override
void Init()
{
    buildIn =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 2,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{

Notification.Print(buildIn.Get
Value().ToString());
}
}
```

See Also

[Reference](#)

[IndicatorsManager Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorsManager Methods

The [IndicatorsManager](#) type exposes the following members.

▪ Methods

Name	Description
  CreateInstance(IndicatorBuilder, Object, HistoricalData)	Returns specific indicator by name from the script lookup
  CreateInstance(String, Object, HistoricalData)	Returns specific indicator by name from script lookup
  Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
  GetHashCode	Serves as the default hash function.

(Inherited from [Object](#))



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

▲ See Also

[Reference](#)

[IndicatorsManager Class](#)

[TradeApi.Indicators Namespace](#)

IndicatorsManagerCreate Instance Method

▪ Overload List

Name	Description
  CreateInstance(IndicatorBuilder, Object, HistoricalData)	Returns specific indicator by name from the script lookup
  CreateInstance(String, Object, HistoricalData)	Returns specific indicator by name from script lookup

[Top](#)

▪ See Also

[Reference](#)

[IndicatorsManager Class](#)

[TradeApi.Indicators Namespace](#)

Indicators ManagerCreate Instance(IndicatorBuilder, Object, HistoricalData) Method

Returns specific indicator by name from the script lookup

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public BuiltInIndicator  
CreateInstance(  
    IndicatorBuilder  
indicatorBuilder,  
    Object[] parameters,  
    HistoricalData data  
)
```

[Copy](#)

Parameters

indicatorBuilder [IndicatorBuilder](#)

An instance of the platform indicator class

parameters Object

A key value set of the params of an indicator

data HistoricalData

The indicator's based on historical data

Return Value

BuiltInIndicator

Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
CreateInstanceExample : IndicatorBuilder
    {
        public
CreateInstanceExample()
            : base()
        {
            #region
Initialization

Credentials.ProjectName =
"CreateInstanceExample";
            #endregion

Lines.Set("CreateInstanceExamp
le");

base.SeparateWindow = false;
    }
```

```
[InputParameter(InputType.Nume
ric, "Period", 0)]
[SimpleNumeric(1D,
99999D)]
public int Period =
13;

        BuiltInIndicator
custom;

        public override
void Init()
{
    custom =
IndicatorsManager.CreateInstance(this, new object[] {
        "MAPeriod",
5}, HistoryDataSeries);
}

        public override
void Update(TickStatus args)
{

Notification.Print($"Current
instance:
{Period.ToString() }");

Lines["CreateInstanceExample"]
.SetValue(Period);

Notification.Print($"Current
instance:
{custom.GetValue().ToString() }
");
}
}
```

See Also

Reference

[IndicatorsManager Class](#)

[CreateInstance Overload](#)

[TradeApi.Indicators Namespace](#)

Indicators ManagerCreate Instance(String, Object, HistoricalData) Method

Returns specific indicator by name from script lookup

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BuiltInIndicator  
CreateInstance(  
    string IndicatorClass,  
    Object[] parameters,  
    HistoricalData data  
)
```

[Copy](#)

Parameters

IndicatorClass [String](#)

The name of the indicator's class

parameters [Object](#)

A key value set of the params of an indicator

data [HistoricalData](#)

The indicator's based on historical data

Return Value
[BuiltInIndicator](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
CreateInstanceExample : 
IndicatorBuilder
    {
        public
CreateInstanceExample()
            : base()
        {
            #region
Initialization

Credentials.ProjectName =
"CreateInstanceExample";
            #endregion

Lines.Set("CreateInstanceExamp
le");

base.SeparateWindow = false;
    }

    [InputParameter(InputType.Nume
ric, "Period", 0)]
}
```

```
        [SimpleNumeric(1D,
99999D)]
    public int Period =
13;

        BuiltInIndicator
custom;

    public override
void Init()
{
    custom =
IndicatorsManager.CreateInstance("CreateInstanceExample",
new object[] {
                    "MAPeriod",
5}, HistoryDataSeries);
}

    public override
void Update(TickStatus args)
{

Notification.Print($"Current
instance:
{Period.ToString() }");

Lines["CreateInstanceExample"]
.SetValue(Period);

Notification.Print($"Custom
instance:
{custom.GetValue().ToString() }
");
}
}
```

See Also

Reference

[IndicatorsManager Class](#)

[CreateInstance Overload](#)

[TradeApi.Indicators Namespace](#)

Lines Class

Lines entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.Indicators](#) Lines

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class Lines
```

[Copy](#)

The [Lines](#) type exposes the following members.

► Properties

	Name	Description
	Count	Lines count
	ItemInt32	Lines indexer
	ItemString	Lines indexer

[Top](#)

► Methods

	Name	Description
--	------	-------------

 	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
  	Set	Setting line into lines series
 	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

▲ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

Lines Properties

The [Lines](#) type exposes the following members.

Properties

	Name	Description
 	Count	Lines count
 	ItemInt32	Lines indexer
 	ItemString	Lines indexer

[Top](#)

See Also

[Reference](#)

[Lines Class](#)

[TradeApi.Indicators Namespace](#)

LinesCount Property

Lines count

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

Int32

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
CountExample :
IndicatorBuilder
{
    public
CountExample()
        : base()
    {
        #region
Initialization
    }
}
```

[Copy](#)

```
Credentials.ProjectName =  
    "CountExample";  
    #endregion  
  
    Lines.Set("LinesExample");  
  
    base.SeparateWindow = false;  
}  
  
    public override  
void Init()  
{  
  
    Notification.Print(Lines.Count  
.ToString());  
}  
  
    public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

Reference

[Lines Class](#)

[TradeApi.Indicators Namespace](#)

LinesItem Property

▪ Overload List

Name	Description
 ItemInt32	Lines indexer
 ItemString	Lines indexer

[Top](#)

▪ See Also

[Reference](#)

[Lines Class](#)

[TradeApi.Indicators Namespace](#)

LinesItem(Int32) Property

Lines indexer

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public IndicatorLine this[  
    int index  
] { get; }
```

[Copy](#)

Parameters

index Int32

Property Value

IndicatorLine

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
LinesExample :
IndicatorBuilder
{
    public
LinesExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"LinesExample";
        #endregion

        Lines.Set("LinesExample");
    }

    public override
void Init()
{
    Lines[0].Color
= Color.Orange;
}

    public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[Lines Class](#)

[Item Overload](#)

[TradeApi.Indicators Namespace](#)

LinesItem(String) Property

Lines indexer

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public IndicatorLine this[  
    string name  
] { get; }
```

[Copy](#)

Parameters

name [String](#)

Property Value

[IndicatorLine](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
LinesExample :  
IndicatorBuilder  
{  
    public  
    LinesExample()  
        : base()  
    {  
        #region  
        Initialization  
  
        Credentials.ProjectName =  
        "LinesExample";  
        #endregion  
  
        Lines.Set("LinesExample");  
  
        base.SeparateWindow = false;  
    }  
  
    public override  
    void Init()  
    {  
  
        Lines["LinesExample"].Color =  
        Color.Orange;  
    }  
  
    public override  
    void Update(TickStatus args)  
    {  
  
    }  
}
```

See Also

Reference

[Lines Class](#)

[Item Overload](#)

[TradeApi.Indicators Namespace](#)

Lines Methods

The [Lines](#) type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	Set	Setting line into lines series
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[Lines Class](#)

[TradeApi.Indicators Namespace](#)

LinesSet Method

Setting line into lines series

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public void Set(Copy
                 string name
               )
```

Parameters

name String
line name

◀ Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class SetExample
        : IndicatorBuilder
    {
        public SetExample()
            : base()
```

```
        {
            #region
            Initialization

            Credentials.ProjectName =
            "SetExample";
            #endregion

            Lines.Set("LinesExample");

            base.SeparateWindow = false;
        }

        public override
void Init()
{
    Lines["LinesExample"].Color =
    Color.Orange;
}

        public override
void Update(TickStatus args)
{
}

}

}
```

See Also

[Reference](#)

[Lines Class](#)

[TradeApi.Indicators Namespace](#)

LineStyle Enumeration

Lines style types

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public enum LineStyle
```

[Copy](#)

◀ Members

Member name	Value	Description
Solid	0	Solid
Dot	1	Dot
DotLine	2	DotLine
DotSpaced	5	DotSpaced
DotBigSpaced	6	DotBigSpaced
Histogram	7	Histogram
Symbol	8	Symbol
Horizontal	9	Horizontal
Ladder	10	Ladder

See Also

Reference

[TradeApi.Indicators Namespace](#)

MAMode Enumeration

Moving average mode type

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum MAMode
```

[Copy](#)

► Members

Member name	Value	Description
SMA	0	Simple moving average calculation
EMA	1	Exponential moving average calculation
SMMA	2	Smoothed moving average calculation
LWMA	3	Linear weighted moving average calculation

► See Also

Reference

TradeApi.Indicators Namespace

Markers Class

Markers entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsMarke
rs](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class Markers
```

[Copy](#)

The [Markers](#) type exposes the following members.

▪ Methods

	Name	Description
	Clear	Clears marker
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)



[GetHashCode](#) Serves as the default hash function.
(Inherited from [Object](#))



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



[Set](#)

Sets marker



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

Markers Methods

The [Markers](#) type exposes the following members.

▪ Methods

	Name	Description
 	Clear	Clears marker
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
 	Set	Sets marker
	ToString	Returns a string that represents the

current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[Markers Class](#)

[TradeApi.Indicators Namespace](#)

MarkersClear Method

Clears marker

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public void Clear()
```

[Copy](#)

◀ Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
ClearExample :
IndicatorBuilder
{
    public
ClearExample()
        : base()
    {
        #region
Initialization

        Credentials.ProjectName =
    }
}
```

[Copy](#)

```
"ClearExample";
    #endregion

Lines.Set("MarkersExample");

base.SeparateWindow = false;
}

        public override
void Init()
{
    Lines["MarkersExample"].Markers.Set(Color.Orange, 0);
}

        public override
void Update(TickStatus args)
{
    Lines["MarkersExample"].Markers.Clear();
}
}
```

See Also

Reference

[Markers Class](#)

[TradeApi.Indicators Namespace](#)

MarkersSet Method

Sets marker

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public void Set(Copy
    Color color,
    int offset = 0
)
```

Parameters

color Color

offset Int32 (Optional)

◀ Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class SetExample
        : IndicatorBuilder
```

```
    {
        public SetExample()
        : base()
        {
            #region Initialization
            Credentials.ProjectName =
            "SetExample";
            #endregion
        }

        Lines.Set("MarkersExample");
    }

    base.SeparateWindow = false;
}

public override void Init()
{
    Lines["MarkersExample"].Markers.Set(Color.Orange, 0);
}

public override void Update(TickStatus args)
{
}

}

}
```

See Also

[Reference](#)

[Markers Class](#)

[TradeApi.Indicators Namespace](#)

PrimitiveFigure Enumeration

Primitive figure type

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum PrimitiveFigure
```

[Copy](#)

► Members

Member name	Value	Description
Rectangle	0	Rectangle
Circle	1	Circle
Diamond	2	Diamond
TriangleUp	3	TriangleUp
TriangleDown	4	TriangleDown

► See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

RSIMode Enumeration

RSI mode type

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum RSIMode
```

[Copy](#)

► Members

Member name	Value	Description
Simple	0	Simple moving average RSI calculation
Exponential	1	Exponential RSI calculation

► See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

Shift Enumeration

Supported line's shift

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum Shift
```

[Copy](#)

► Members

Member name	Value	Description
ToBack	0	ToBack
ToFront	1	ToFront

► See Also

[Reference](#)

[TradeApi.Indicators Namespace](#)

Threshold Class

Threshold entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsThreshold](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class Threshold
```

[Copy](#)

The `Threshold` type exposes the following members.

▪ Properties

	Name	Description
	Color	Threshold color
	Level	Threshold level
	Name	Threshold name
	Style	Threshold line style
	Visible	Threshold visibility

Width Threshold width

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

Reference

TradeApi.Indicators Namespace

Threshold Properties

The [Threshold](#) type exposes the following members.

Properties

	Name	Description
 	Color	Threshold color
 	Level	Threshold level
 	Name	Threshold name
 	Style	Threshold line style
 	Visible	Threshold visibility
 	Width	Threshold width

[Top](#)

See Also

[Reference](#)

[Threshold Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdColor Property

Threshold color

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Color Color { get; set; } Copy
```

Property Value

Color

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class
ColorExample :
IndicatorBuilder
{
    public
ColorExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ColorExample";
        #endregion

        Thresholds.Set("ThresholdsExam
        ple");

        base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        Thresholds["ThresholdsExample"]
        .Color = Color.Orange;
    }

        public override
void Update(TickStatus args)
{
}

    }
}
```

See Also

Reference

[Threshold Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdLevel Property

Threshold level

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Level { get;  
    set; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
LevelExample :  
IndicatorBuilder  
    {  
        public  
LevelExample()  
            : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "LevelExample";
            #endregion

            Thresholds.Set("ThresholdsExam
ple");

            base.SeparateWindow = false;
        }

        public override
void Init()
{
    Thresholds["ThresholdsExample"]
        .Level = 80;
}

        public override
void Update(TickStatus args)
{
}

}

}
```

See Also

[Reference](#)

[Threshold Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdName Property

Threshold name

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Name { get; set; } Copy
```

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
NameExample : IndicatorBuilder
    {
        public
NameExample()
            : base()
        {

```

```
        #region
Initialization

Credentials.ProjectName =
"NameExample";
        #endregion

Thresholds.Set("ThresholdsExam
ple");

base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{

Notification.Print(Thresholds[
"ThresholdsExample"].Name.ToString());
}
}
}
```

See Also

[Reference](#)

[Threshold Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdStyle Property

Threshold line style

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public LineStyle Style { get;  
    set; }
```

[Copy](#)

Property Value

[LineStyle](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
StyleExample :  
IndicatorBuilder  
    {  
        public  
StyleExample()  
            : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "StyleExample";
            #endregion

            Thresholds.Set("ThresholdsExam
                            ple");

            base.SeparateWindow = false;
        }

        public override
void Init()
{
    Thresholds["ThresholdsExample"]
        .Style = LineStyle.Dot;
}

        public override
void Update(TickStatus args)
{
}

}

}
```

See Also

[Reference](#)

[Threshold Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdVisible Property

Threshold visibility

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool Visible { get;  
    set; }
```

[Copy](#)

Property Value

[Boolean](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
VisibleExample :  
IndicatorBuilder  
    {  
        public  
VisibleExample()  
            : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "VisibleExample";  
    #endregion  
  
    Threshold.Set("ThresholdExample");  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
    Threshold["ThresholdExample"].  
    Visible = true;  
}  
  
public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

[Reference](#)

[Threshold Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdWidth Property

Threshold width

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Width { get; set; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
WidthExample :
IndicatorBuilder
{
    public
WidthExample()
    : base()
    {
```

```
        #region
Initialization

Credentials.ProjectName =
"WidthExample";
        #endregion

Thresholds.Set("ThresholdsExam
ple");

base.SeparateWindow = false;
    }

        public override
void Init()
{
    Thresholds["ThresholdsExample"]
.Width = 2;
}

        public override
void Update(TickStatus args)
{
    }
}
```

See Also

Reference

[Threshold Class](#)

[TradeApi.Indicators Namespace](#)

Threshold Methods

The [Threshold](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[Threshold Class](#)

[TradeApi.Indicators Namespace](#)

Thresholds Class

Thresholds entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsThresholds](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class Thresholds Copy
```

The [Thresholds](#) type exposes the following members.

▪ Properties

	Name	Description
	Count	Thresholds count
	ItemInt32	Thresholds indexer
	ItemString	Thresholds indexer

[Top](#)

◀ Methods

	Name	Description
≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetType	Gets the Type of the current instance. (Inherited from Object)
≡ ⚡	Set	Threshold set
≡	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

Reference

[TradeApi.Indicators Namespace](#)

Thresholds Properties

The [Thresholds](#) type exposes the following members.

Properties

	Name	Description
 	Count	Thresholds count
 	ItemInt32	Thresholds indexer
 	ItemString	Thresholds indexer

[Top](#)

See Also

[Reference](#)

[Thresholds Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdsCount Property

Thresholds count

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
CountExample :
IndicatorBuilder
{
    public
CountExample()
    : base()
{
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"CountExample";
        #endregion

Thresholds.Set("ThresholdsExam
ple");

base.SeparateWindow = false;
    }

        public override
void Init()
{
}

Notification.Print(Thresholds.
Count.ToString());
    }

        public override
void Update(TickStatus args)
{
}

}
}
```

See Also

Reference

[Thresholds Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdsItem Property

▪ Overload List

	Name	Description
	ItemInt32	Thresholds indexer
	ItemString	Thresholds indexer

[Top](#)

▪ See Also

[Reference](#)

[Thresholds Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdsItem(Int32) Property

Thresholds indexer

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Threshold this[  
    int index  
] { get; }
```

[Copy](#)

Parameters

index [Int32](#)
threshold index

Property Value
[Threshold](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
ThresholdsExample :
IndicatorBuilder
{
    public
ThresholdsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ThresholdsExample";
    #endregion

Thresholds.Set("ThresholdsExam
ple");

base.SeparateWindow = false;
}

    public override
void Init()
{
    Thresholds[0].Color =
Color.Orange;
}

    public override
void Update(TickStatus args)
{
}

}
}
```

See Also

[Reference](#)

[Thresholds Class](#)

[Item Overload](#)

[TradeApi.Indicators Namespace](#)

ThresholdsItem(String) Property

Thresholds indexer

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Threshold this[  
    string name  
] { get; }
```

[Copy](#)

Parameters

name [String](#)
threshold name

Property Value

[Threshold](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
ThresholdsExample :
IndicatorBuilder
{
    public
ThresholdsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ThresholdsExample";
    #endregion

Thresholds.Set("ThresholdsExam
ple");

base.SeparateWindow = false;
}

    public override
void Init()
{
    Thresholds["ThresholdsExample"]
.Color = Color.Orange;
}

    public override
void Update(TickStatus args)
{
}

}
}
```

See Also

[Reference](#)

[Thresholds Class](#)

[Item Overload](#)

[TradeApi.Indicators Namespace](#)

Thresholds Methods

The [Thresholds](#) type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	Set	Threshold set
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[Thresholds Class](#)

[TradeApi.Indicators Namespace](#)

ThresholdsSet Method

Threshold set

Namespace: TradeApi.Indicators

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void Set(Copy
                 string name
               )
```

Parameters

name String
threshold name

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using System.Drawing;

namespace TestEnv
{
    public class SetExample
        : IndicatorBuilder
    {
        public SetExample()
            : base()
```

```
        {
            #region
            Initialization

            Credentials.ProjectName =
            "SetExample";
            #endregion

            Thresholds.Set("ThresholdsExam
            ple");
        }

        base.SeparateWindow = false;
    }

    public override
    void Init()
    {
        Thresholds["ThresholdsExample"]
        .Color = Color.Orange;
    }

    public override
    void Update(TickStatus args)
    {
    }
}
```

See Also

Reference

[Thresholds Class](#)

[TradeApi.Indicators Namespace](#)

TimeValue Structure

TimeValue entity

► Inheritance Hierarchy

[SystemObject](#) [SystemValueType](#)
[TradeApi.Indicators](#)[TimeValue](#)

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public struct TimeValue
```

[Copy](#)

The [TimeValue](#) type exposes the following members.

► Constructors

	Name	Description
	TimeValue	Constructor

[Top](#)

► Properties

	Name	Description
 	Empty	Empty time value
 		

[TimeUtc](#) UTC time of the unit is dispatched



[Value](#)

UTC time of the unit is dispatched

[Top](#)

Methods

	Name	Description
	Equals	(Overrides ValueTypeEquals(Object))
	GetHashCode	(Overrides ValueTypeGetHashCode)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType)

[Top](#)

Operators

	Name	Description
	Equality(TimeValue, TimeValue)	
	Inequality(TimeValue,	

TimeValue)

[Top](#)

◀ See Also

Reference

[TradeApi.Indicators Namespace](#)

TimeValue Constructor

Cunstructor

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public TimeValue(Copy
                  DateTime timeUTC,
                  double value
                )
```

Parameters

timeUTC [DateTime](#)

value [Double](#)

▪ Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using Runtime.Script.Charts;
using System.Drawing;

namespace TestEnv
```

```
    {
        public class
TimeValueExample : 
IndicatorBuilder
{
    public
TimeValueExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"TimeValueExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
}

        public override
void OnPaintChart(object
sender, PaintEventArgs
args)
{
    var mainWidow
=
ChartSource.GetAllWindows().Fi
nd(win => win.IsMainWindow);
```

```
        var
windowRectUpperPoint =
mainWidow.WindowRectangle.Location;
        var
windowHighestPriceEdge =
ChartSource.PointToChart(windo
wRectUpperPoint);
        var
timeValueHighest =
ChartSource.GetTimeValue(windo
wHighestPriceEdge,
mainWidow.WindowNumber);

timeValueHighest = new
TimeValue(timeValueHighest.Tim
eUtc.AddMinutes(5),
timeValueHighest.Value);

        PointF
dividerLineStart =
ChartSource.GetChartPoint(time
ValueHighest,
mainWidow.WindowNumber);

        var
windowLowestPriceEdge =
ChartSource.PointToChart(new
Point(mainWidow.WindowRectangl
e.Left,
mainWidow.WindowRectangle.Bott
om));
        var
timeValueLowest =
ChartSource.GetTimeValue(windo
wLowestPriceEdge,
mainWidow.WindowNumber);

timeValueLowest = new
TimeValue(timeValueLowest.Time
Utc.AddMinutes(5),
```

```
    timeValueLowest.Value);
    PointF
dividerLineEnd =
ChartSource.GetChartPoint(time
ValueLowest,
mainWidow.WindowNumber);

args.Graphics.DrawLine(new
Pen(Color.Orange, 2),
dividerLineStart,
dividerLineEnd);

ChartSource.GUIRefresh();
}
}
}
```

See Also

Reference

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValue Properties

The [TimeValue](#) type exposes the following members.

Properties

	Name	Description
 	Empty	Empty time value
 	TimeUtc	UTC time of the unit is dispatched
 	Value	UTC time of the unit is dispatched

[Top](#)

See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValueEmpty Property

Empty time value

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static TimeValue Empty
{ get; }
```

[Copy](#)

Property Value

[TimeValue](#)

► See Also

Reference

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValueTimeUtc Property

UTC time of the unit is dispatched

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public DateTime TimeUtc {  
    get;  
}
```

[Copy](#)

Property Value
[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using System.Windows.Forms;  
  
namespace TestEnv  
{  
    public class  
TimeUtcExample :  
IndicatorBuilder  
    {
```

[Copy](#)

```
        public
TimeUtcExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TimeUtcExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    Point
cursorPos =
ChartSource.PointToChart(Curso
r.Position);
    var timeValue
=
ChartSource.GetTimeValue(curso
rPos,
ChartSource.GetAllWindows().Fi
nd(win =>
win.IsMainWindow).WindowNumber
);

Notification.Print(timeValue.T
imeUtc.ToString("yyyy-MM-dd HH:mm:ss"));
}
}
```

See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValueValue Property

UTC time of the unit is dispatched

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Value {  
    get;  
}
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using System.Windows.Forms;  
  
namespace TestEnv  
{  
    public class  
TimeUtcExample :  
IndicatorBuilder  
    {
```

[Copy](#)

```
    public
TimeUtcExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TimeUtcExample";
    #endregion
}

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    Point
cursorPos =
ChartSource.PointToChart(Curso
r.Position);
    var timeValue
=
ChartSource.GetTimeValue(curso
rPos,
ChartSource.GetAllWindows().Fi
nd(win =>
win.IsMainWindow).WindowNumber
);

Notification.Print(timeValue.V
alue.ToString());
}
}
```

See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValue Methods

The [TimeValue](#) type exposes the following members.

► Methods

Name	Description
 Equals	(Overrides ValueType.Equals(Object))
 GetHashCode	(Overrides ValueType.GetHashCode)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType)

[Top](#)

► See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValueEquals Method

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public override bool Equals(Copy
    Object obj
)
```

Parameters

obj Object

Return Value

Boolean

◀ See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValueGetHash

Code Method

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override int  
GetHashCode()
```

[Copy](#)

Return Value

[Int32](#)

► See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValue Operators

The [TimeValue](#) type exposes the following members.

Operators

Name	Description
 S	Equality(TimeValue, TimeValue)
 S	Inequality(TimeValue, TimeValue)

[Top](#)

See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValueEquality Operator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public static bool operator ==  
(  
    TimeValue timeValue1,  
    TimeValue timeValue2  
)
```

[Copy](#)

Parameters

timeValue1 [TimeValue](#)

timeValue2 [TimeValue](#)

Return Value

[Boolean](#)

◀ See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

TimeValueInequality Operator

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public static bool operator != Copy
(
    TimeValue timeValue1,
    TimeValue timeValue2
)
```

Parameters

timeValue1 [TimeValue](#)

timeValue2 [TimeValue](#)

Return Value

[Boolean](#)

◀ See Also

[Reference](#)

[TimeValue Structure](#)

[TradeApi.Indicators Namespace](#)

Window Class

Window entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.IndicatorsWindow](#)

Namespace: [TradeApi.Indicators](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

▪ Syntax

C#

```
public sealed class Window
```

[Copy](#)

The [Window](#) type exposes the following members.

▪ Properties

	Name	Description
	IsMainWindow	Checks if current window is main
	WindowNumber	Returns a chart window number



[WindowRectangle](#) Provides a chart area rectangle

[Top](#)

◀ Methods

	Name	Description
≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetType	Gets the Type of the current instance. (Inherited from Object)
≡	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

Reference

[TradeApi.Indicators Namespace](#)

Window Properties

The [Window](#) type exposes the following members.

► Properties

	Name	Description
 	IsMainWindow	Checks if current window is main
 	WindowNumber	Returns a chart window number
 	WindowRectangle	Provides a chart area rectangle

[Top](#)

► See Also

[Reference](#)

[Window Class](#)

[TradeApi.Indicators Namespace](#)

WindowIsMainWindow Property

Checks if current window is main

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool IsMainWindow {  
    get; }
```

[Copy](#)

Property Value

[Boolean](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System.Drawing;  
using System.Windows.Forms;  
  
namespace TestEnv  
{  
    public class  
        IsMainWindowExample :  
        IndicatorBuilder  
    {  
        public
```

[Copy](#)

```
    IsMainWindowExample()
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"IsMainWindowExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    var winNumber =
    ChartSource.GetAllWindows().Find(win =>
    win.IsMainWindow).WindowNumber
;
    Point
cursorPos =
ChartSource.ChartPointToScreen
(Cursor.Position);
    var timeValue
=
ChartSource.GetTimeValue(cursorPos,
winNumber);

Notification.Print(timeValue.V
alue.ToString());
}
}
```

See Also

[Reference](#)

[Window Class](#)

[TradeApi.Indicators Namespace](#)

WindowWindow Number Property

Returns a chart window number

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int WindowNumber { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using System.Drawing;
using System.Windows.Forms;

namespace TestEnv
{
    public class
WindowNumberExample :
IndicatorBuilder
{
    public
```

```
WindowNumberExample()
    : base()
{
    #region
    Initialization

    Credentials.ProjectName =
    "WindowNumberExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    var winNumber
=
    ChartSource.GetAllWindows().Fi
nd(win =>
    win.IsMainWindow).WindowNumber
;

    Point
cursorPos =
    ChartSource.ChartPointToScreen
    (Cursor.Position);
    var timeValue
=
    ChartSource.GetTimeValue(curso
rPos, winNumber);

    Notification.Print(timeValue.V
alue.ToString());
}
}
```

See Also

[Reference](#)

[Window Class](#)

[TradeApi.Indicators Namespace](#)

WindowWindow Rectangle Property

Provides a chart area rectangle

Namespace: [TradeApi.Indicators](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Rectangle  
WindowRectangle { get; }
```

[Copy](#)

Property Value

[Rectangle](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using Runtime.Script.Charts;  
using System.Drawing;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
WindowRectangleExample :  
IndicatorBuilder  
    {  
        public
```

```
WindowRectangleExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"WindowRectangleExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
}

        public override
void OnPaintChart(object
sender, PaintEventArgs
args)
{
    var mainWidow
=
ChartSource.GetAllWindows().Fi
nd(win => win.IsMainWindow);

    var
windowRectUpperPoint =
mainWidow.WindowRectangle.Loca
tion;
    var
windowHighestPriceEdge =
ChartSource.PointToChart(windo
```

```
wRectUpperPoint);  
        var  
timeValueHighest =  
ChartSource.GetTimeValue(windo  
wHighestPriceEdge,  
mainWidow.WindowNumber);  
        PointF  
dividerLineStart =  
ChartSource.GetChartPoint(time  
ValueHighest,  
mainWidow.WindowNumber);  
  
        var  
windowLowestPriceEdge =  
ChartSource.PointToChart(new  
Point(mainWidow.WindowRectangl  
e.Left,  
mainWidow.WindowRectangle.Bott  
om));  
        var  
timeValueLowest =  
ChartSource.GetTimeValue(windo  
wLowestPriceEdge,  
mainWidow.WindowNumber);  
        PointF  
dividerLineEnd =  
ChartSource.GetChartPoint(time  
ValueLowest,  
mainWidow.WindowNumber);  
  
args.Graphics.DrawLine(new  
Pen(Color.Orange, 2),  
dividerLineStart,  
dividerLineEnd);  
  
ChartSource.GUIRefresh();  
    }  
}
```

See Also

[Reference](#)

[Window Class](#)

[TradeApi.Indicators Namespace](#)

Window Methods

The [Window](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[Window Class](#)

[TradeApi.Indicators Namespace](#)

TradeApi.Instruments Namespace

Classes

Class	Description
BondsInstrument	Bond instrument entity
CFDInstrument	CFD instrument entity
CorporateInstrument	
CryptoInstrument	Crypto instrument entity
DayInfo	Instrument's day info entity
DerivativeContract	DerivativeContract entity
EquitiesInstrument	Equities instrument entity
ETFInstrument	ETF instrument entity
ForexInstrument	Forex instrument entity
ForwardInstrument	Forward instrument entity
FuturesInstrument	Futures instrument entity
IndicesInstrument	Indices instrument entity
Instrument	Instrument entity
InstrumentsManager	InstrumentsManager entity
MarketDepth	Instrument's market depth entity
OptionsInstrument	Options instrument entity
SpotInstrument	Spot instrument entity
SpreadBetInstrument	SpreadBet instrument entity
SyntheticInstrument	Synthetic instrument entity

	SyntheticInstrumentContent	SyntheticInstrumentContent entity
	TBillsInstrument	TBill instrument entity
	TickInterval	TickInterval entity
	Ticksize	Ticksized entity
	TicksizeTickcost	TicksizedTickcost entity

▪ Enumerations

	Enumeration	Description
	DepthAggregate	Market depth aggregate type
	InstrumentType	Supported instrument type
	OptionType	Option type
	TradingStatus	Trading status type

BondsInstrument Class

Bond instrument entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsBondsInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
BondsInstrument : Instrument
```

[Copy](#)

The [BondsInstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)

	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Symbol	Symbol of the base instrument (Inherited from Instrument)
	TradingStatus	Current trading status for the

instrument
(Inherited from
[Instrument](#))

 	Type	Market category of the instrument (Overrides InstrumentType)
---	----------------------	---

[Top](#)

◀ Methods

	Name	Description
 	Equals	Allow to compare two instruments (Inherited from Instrument)
 	GetHashCode	(Inherited from Instrument)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
 	MinimalLot	The smallest

trade allowed
(in lot)
(Inherited from
[Instrument](#))



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

BondsInstrument Properties

The [BondsInstrument](#) type exposes the following members.

Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)

 MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 Symbol	Symbol of the base instrument (Inherited from Instrument)
 TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

◀ See Also

[Reference](#)

[BondsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

BondsInstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value
[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
    {
        public
TypeExample()
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"TypeExample";
            #endregion
        }
OrderRequest
request;
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot());
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Bond)
    {

OrdersManager.Send(request);
    }
}
}
```

◀ See Also

Reference

[BondsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

BondsInstrument Methods

The [BondsInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BondsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CFDInstrument Class

CFD instrument entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsCFDInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
CFDInstrument : Instrument
```

[Copy](#)

The [CFDInstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)
 LimitHigh	Gets the high price limit to the

		selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)



Type

Market category
of the
instrument
(Overrides
[InstrumentType](#))

[Top](#)

Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot) (Inherited from Instrument)



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

See Also

Reference

[TradeApi.Instruments Namespace](#)

CFDInstrument Properties

The [CFDInstrument](#) type exposes the following members.

Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)

 MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 Symbol	Symbol of the base instrument (Inherited from Instrument)
 TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

◀ See Also

[Reference](#)

[CFDInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CFDInstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value
[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
    {
        public
TypeExample()
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"TypeExample";
            #endregion
        }
OrderRequest
request;
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot());
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.CFD)
    {

OrdersManager.Send(request);
    }
}
}
```



See Also

[Reference](#)

[CFDInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CFDInstrument Methods

The [CFDInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[CFDInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CorporateInstrument Class

▪ Inheritance Hierarchy

SystemObject TradeApi.InstrumentsInstrument

TradeApi.InstrumentsCorporateInstrument

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
CorporateInstrument :  
Instrument
```

[Copy](#)

The [CorporateInstrument](#) type exposes the following members.

▪ Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)

	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotSize	LotSize
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MaturityDate	Maturity Date
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Symbol	Symbol of the base instrument (Inherited from Instrument)

 TickSize	TickSize
 TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

◀ Methods

Name	Description
 Equals	Allow to compare two instruments (Inherited from Instrument)
 GetHashCode	(Inherited from Instrument)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 MaximalLot	The biggest trade allowed (in lot)

(Inherited from
[Instrument](#))



[MinimalLot](#)

The smallest
trade allowed
(in lot)
(Inherited from
[Instrument](#))



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

CorporateInstrument Properties

The [CorporateInstrument](#) type exposes the following members.

Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotSize	LotSize
 	LotStep	Step of the lot changes

		(Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MaturityDate	Maturity Date
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TickSize	TickSize
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

◀ See Also

Reference

[CorporateInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Corporate InstrumentLotSize Property

LotSize

Namespace: TradeApi.Instruments
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public int LotSize { get; }
```

[Copy](#)

Property Value

Int32

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
LotSizeExample :
StrategyBuilder
{
    public
```

[Copy](#)

```
LotSizeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"LotSizeExample";
    #endregion
}

        public override void
Init()
{
    var instrument =
InstrumentsManager.Current;
    if
(instrument.Type ==
InstrumentType.Corporate)
    {

Notification.Print($"LotSize:
{(instrument as
CorporateInstrument).LotSize}""
);
    }
}

        public override void
Update(TickStatus args)
{
}

}
}
```

See Also

Reference

CorporateInstrument Class
TradeApi.Instruments Namespace

Corporate InstrumentMaturity Date Property

Maturity Date

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime MaturityDate {  
    get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
MaturityDateExample :  
StrategyBuilder
```

[Copy](#)

```
        {
            public
MaturityDateExample()
            : base()
{
    #region
Initialization

Credentials.ProjectName =
"MaturingDateExample";
#endregion
}

public override
void Init()
{
    var instrument
= InstrumentsManager.Current;
    if
(instrument.Type ==
InstrumentType.Corporate)
    {
        DateTime
maturityDate =
(InstrumentsManager.Current as
CorporateInstrument).MaturityD
ate;

Notification.Print($"MaturityD
ate:
{maturityDate.ToString() }");
    }
}

public override
void Update(TickStatus args)
{
}
```

```
}
```

```
}
```

See Also

[Reference](#)

[CorporateInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Corporate InstrumentTickSize Property

TickSize

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Ticksize> TickSize
{ get; }
```

[Copy](#)

Property Value

[ListTicksize](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
    TickSizeExample :
    StrategyBuilder
```

[Copy](#)

```
{  
    public  
    TickSizeExample()  
        : base()  
    {  
        #region  
        Initialization  
  
        Credentials.ProjectName =  
        "TickSizeExample";  
        #endregion  
    }  
  
    public override  
    void Init()  
    {  
        var instrument  
        = InstrumentsManager.Current;  
        if  
(instrument.Type ==  
InstrumentType.Corporate)  
        {  
  
            Notification.Print($"TickSize:  
            {(instrument as  
CorporateInstrument).TickSize}  
");  
        }  
    }  
  
    public override  
    void Update(TickStatus args)  
    {  
  
    }  
}
```

See Also

[Reference](#)

[CorporateInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Corporate InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value

[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
```

[Copy](#)

```
    public
TypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TypeExample";
    #endregion
}
OrderRequest
request;
Instrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Corporate)
{
    OrdersManager.Send(request);
}
```

```
        }  
    }  
}
```

See Also

Reference

[CorporateInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CorporateInstrument Methods

The [CorporateInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[CorporateInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CryptoInstrument Class

Crypto instrument entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsCryptoInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
CryptoInstrument : Instrument
```

[Copy](#)

The [CryptoInstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)

	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotSize	Amount of base asset for one lot
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Precision	Evaluates a precision unit of selected instrument

	Symbol	Symbol of the base instrument (Inherited from Instrument)
	TickSize	Min value size of pair's quote used by script
	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

◀ Methods

Name	Description
 Equals	Allow to compare two instruments (Inherited from Instrument)
 GetHashCode	(Inherited from Instrument)
 GetType	Gets the Type of the current

instance.
(Inherited from
[Object](#))



[MaximalLot](#)

The biggest
trade allowed
(in lot)
(Inherited from
[Instrument](#))



[MinimalLot](#)

The smallest
trade allowed
(in lot)
(Inherited from
[Instrument](#))



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

CryptoInstrument Properties

The [CryptoInstrument](#) type exposes the following members.

Properties

Name	Description
  DayInfo	Day bar information for an instrument (Inherited from Instrument)
  LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
  LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
  LotSize	Amount of base asset for one lot
  LotStep	Step of the lot changes

		(Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Precision	Evaluates a precision unit of selected instrument
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TickSize	Min value size of pair's quote used by script
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument

(Overrides
[InstrumentType](#))

[Top](#)

◀ See Also

[Reference](#)

[CryptoInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CryptoInstrumentLot Size Property

Amount of base asset for one lot

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int LotSize { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
LotSizeExample :
StrategyBuilder
{
    public
LotSizeExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"LotSizeExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
    if (instrument
is CryptoInstrument &&
(instrument as
CryptoInstrument).LotSize > 0)
        request =
new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy, (instrument as
CryptoInstrument).LotSize);
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request != null)
{
```

```
        OrdersManager.Send(request);  
    }  
}  
}  
}
```



▲ See Also

Reference

[CryptoInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Crypto InstrumentPrecision Property

Evaluates a precision unit of selected instrument

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public int Precision { get; } Copy
```

Property Value

Int32

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Trading;
using TradeApi.History;
using System;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
PrecisionExample :
```

```
StrategyBuilder
{
    public
PrecisionExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"PrecisionExample";
    #endregion
}
OrderRequest
request;
CryptoInstrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current as
CryptoInstrument;
    request = new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));

request.SLTPPriceType =
SLTPPriceType.Absolute;

request.StopLossPrice =
Math.Round(instrument.TickSize
, instrument.Precision);
}
```

```
        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.Crypto)
    {

OrdersManager.Send(request);
    }
}
}
```

◀ See Also

Reference

[CryptoInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CryptoInstrumentTick Size Property

Min value size of pair's quote used by script

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double TickSize { get;  
}
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
    TickSizeExample :  
    StrategyBuilder  
    {
```

[Copy](#)

```
        public
    TickSizeExample()
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "TickSizeExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument as
CryptoInstrument;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy,
instrument.LotSize);

    request.SLTPPriceType =
SLTPPriceType.Absolute;

    request.StopLossPrice =
Math.Round(instrument.TickSize
, instrument.Precision);
}

        public override
void Update(TickStatus args)
{
```

```
        if (args ==  
    TickStatus.IsBar && request !=  
    null)  
    {  
  
    OrdersManager.Send(request);  
    }  
    }  
}
```

See Also

Reference

[CryptoInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CryptoInstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value
[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
    public
TypeExample()
```

[Copy](#)

```
        : base()
    {
        #region
Initialization

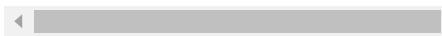
        Credentials.ProjectName =
"TypeExample";
        #endregion
    }
        OrderRequest
request;
        Instrument
instrument;

        public override
void Init()
{
        instrument =
InstrumentsManager.Current;
        request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
        if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.Crypto)
{

OrdersManager.Send(request);
}
```

```
        }  
    }  
}
```



See Also

[Reference](#)

[CryptoInstrument Class](#)

[TradeApi.Instruments Namespace](#)

CryptoInstrument Methods

The [CryptoInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[CryptoInstrument Class](#)

[TradeApi.Instruments Namespace](#)

DayInfo Class

Instrument's day info entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsDayInfo](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class DayInfo
```

[Copy](#)

The [DayInfo](#) type exposes the following members.

► Properties

Name	Description
 Ask	Instrument's ask price
 AskSize	Instrument's ask size value
 Bid	Instrument's bid price

	BidSize	Instrument's bid size value
	Change	Instrument's day change value
	High	Instrument's last high price
	IndAuctionPrice	Instrument's indicative auction price
	Instrument	Instrument's symbol name
	Last	Instrument's last price
	LastBBO	Instrument's best bid offer
	LastSize	Instrument's last size value
	LastTimeUtc	Instrument's last time update in UTC
	LastTrade	Instrument's last trade

	Low	Instrument's last low price
	MatchVolume	Instrument's match volume value
	Open	Instrument's last open price
	OpenInterest	Open interest price
	PastSettlementPrice	Instrument's past settlement price value
	PrevClose	Instrument's previous close price
	SettlementPrice	Instrument's settlement price value
	Ticks	Instrument's ticks
	Volume	Instrument's volume value
Top		

◀ Methods

	Name	Description
≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetType	Gets the Type of the current instance. (Inherited from Object)
≡	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

DayInfo Properties

The [DayInfo](#) type exposes the following members.

Properties

Name	Description
 Ask	Instrument's ask price
 AskSize	Instrument's ask size value
 Bid	Instrument's bid price
 BidSize	Instrument's bid size value
 Change	Instrument's day change value
 High	Instrument's last high price
 IndAuctionPrice	Instrument's indicative auction price

Instrument	Instrument's symbol name
 Last	Instrument's last price
 LastBBO	Instrument's best bid offer
 LastSize	Instrument's last size value
 LastTimeUtc	Instrument's last time update in UTC
 LastTrade	Instrument's last trade
 Low	Instrument's last low price
 MatchVolume	Instrument's match volume value
 Open	Instrument's last open price
 OpenInterest	Open interest price

 	PastSettlementPrice	Instrument's past settlement price value
 	PrevClose	Instrument's previous close price
 	SettlementPrice	Instrument's settlement price value
 	Ticks	Instrument's ticks
 	Volume	Instrument's volume value

[Top](#)

◀ See Also

[Reference](#)

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoAsk Property

Instrument's ask price

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Ask { get; }
```

[Copy](#)

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class AskExample
        : IndicatorBuilder
    {
        public AskExample()
            : base()
        {
            #region
            Initialization
            Credentials.ProjectName =
        }
    }
}
```

[Copy](#)

```
"AskExample";
    #endregion

    base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

    Notification.Comment($"Instrument {instrument.Symbol} ask
price is
{instrument.DayInfo.Ask}");
}
}
```

See Also

[Reference](#)

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoAskSize Property

Instrument's ask size value

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double AskSize { get; } Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
AskSizeExample :
IndicatorBuilder
{
    public
AskSizeExample()
        : base()
    {

```

```
        #region
Initialization

Credentials.ProjectName =
"AskSizeExample";
        #endregion

base.SeparateWindow = false;
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

Notification.Comment($"Instrument {instrument.Symbol} ask
size is
{instrument.DayInfo.AskSize}")
;
}
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoBid Property

Instrument's bid price

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Bid { get; }
```

[Copy](#)

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class BidExample
        : IndicatorBuilder
    {
        public BidExample()
            : base()
        {
            #region
            Initialization
            Credentials.ProjectName =
        }
    }
}
```

[Copy](#)

```
"BidExample";
    #endregion

    base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

    Notification.Comment($"Instrument {instrument.Symbol} bid
price is
{instrument.DayInfo.Bid}");
}
}
```

See Also

[Reference](#)

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoBidSize Property

Instrument's bid size value

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double BidSize { get; } Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
BidSizeExample :
IndicatorBuilder
{
    public
BidSizeExample()
    : base()
    {
```

```
        #region
Initialization

Credentials.ProjectName =
"BidSizeExample";
        #endregion

base.SeparateWindow = false;
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

Notification.Comment($"Instrument {instrument.Symbol} bid
size is
{instrument.DayInfo.BidSize}")
;
}
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoChange Property

Instrument's day change value

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Change { get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
ChangeExample :
IndicatorBuilder
{
    public
ChangeExample()
    : base()
{
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"ChangeExample";
        #endregion

base.SeparateWindow = false;
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

Notification.Comment($"Instrument {instrument.Symbol} day
change is
{instrument.DayInfo.Change}");
}
}
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoHigh Property

Instrument's last high price

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double High { get; }
```

[Copy](#)

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
HighExample : IndicatorBuilder
    {
        public
HighExample()
        : base()
        {
            #region
Initialization
        }
    }
}
```

[Copy](#)

```
Credentials.ProjectName =  
    "HighExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
    public override  
void Init()  
{  
  
}  
  
    public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
Notification.Comment($"Instrument {instrument.Symbol} high  
price is  
{instrument.DayInfo.High}");  
}  
}  
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoIndAuction Price Property

Instrument's indicative auction price

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double IndAuctionPrice
{ get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
IndAuctionPriceExample :
IndicatorBuilder
{
    public
IndAuctionPriceExample()
    : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "IndAuctionPriceExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
}  
  
public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
    Notification.Comment($"Instrument {instrument.Symbol}  
indicative auction price is  
{instrument.DayInfo.IndAuction  
Price}");  
}  
}  
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoInstrument Property

Instrument's symbol name

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Instrument {  
    get; }
```

[Copy](#)

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
Instrumentxample :  
IndicatorBuilder  
    {  
        public  
Instrumentxample()  
        : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
"Instrumentxample";
            #endregion

            base.SeparateWindow = false;
        }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

    Notification.Comment($"Current
instrument symbol
{instrument.DayInfo.Instrument
}");
}
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoLast Property

Instrument's last price

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Last { get; }
```

[Copy](#)

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
LastExample : IndicatorBuilder
    {
        public
LastExample()
        : base()
        {
            #region
Initialization
        }
    }
}
```

[Copy](#)

```
Credentials.ProjectName =  
    "LastExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
    public override  
void Init()  
{  
  
}  
  
    public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
Notification.Comment($"Instrument {instrument.Symbol} last  
price is  
{instrument.DayInfo.Last}");  
}  
}  
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoLastBBO Property

Instrument's best bid offer

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public BBO LastBBO { get; }
```

[Copy](#)

Property Value

BBO

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
LastBBOExample :
IndicatorBuilder
{
    public
LastBBOExample()
    : base()
{
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"LastBBOExample";
        #endregion

base.SeparateWindow = false;
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

Notification.Comment($"Instrument {instrument.Symbol} best
bid offer is
{instrument.DayInfo.LastBBO.Bid}");
}
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoLastSize Property

Instrument's last size value

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double LastSize { get; }Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
LastSizeExample :
IndicatorBuilder
{
    public
LastSizeExample()
    : base()
```

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "LastSizeExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
}  
  
public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
    Notification.Comment($"Instrument {instrument.Symbol} last  
size is  
{instrument.DayInfo.LastSize}"  
);  
    }  
}  
}
```

See Also

[Reference](#)

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoLastTimeUtc Property

Instrument's last time update in UTC

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime LastTimeUtc {  
    get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using System;  
  
namespace TestEnv  
{  
    public class  
TicksExample :  
IndicatorBuilder  
    {  
        public  
TicksExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "TicksExample";
        #endregion

        base.SeparateWindow = false;
    }

        DateTime
prevLastTimeUTC;

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

    if(prevLastTimeUTC==null ||
prevLastTimeUTC
>instrument.DayInfo.LastTimeUt
c)
{
    prevLastTimeUTC
=
instrument.DayInfo.LastTimeUtc
;
}

Notification.Comment($"Instrum
ent {instrument.Symbol} time
```

```
update is
{instrument.DayInfo.LastTimeUTC}");  
    }  
}  
}  
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoLastTrade Property

Instrument's last trade

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Trade LastTrade { get; } Copy
```

Property Value

[Trade](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
LastTradeExample : IndicatorBuilder
    {
        public
LastTradeExample()
            : base()
```

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "LastTradeExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
}  
  
public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
    Notification.Comment($"Instrument {instrument.Symbol} last  
trade price is  
{instrument.DayInfo.LastTrade.  
Price}");  
}  
}  
}
```

See Also

[Reference](#)

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoLow Property

Instrument's last low price

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Low { get; }
```

[Copy](#)

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class LowExample
        : IndicatorBuilder
    {
        public LowExample()
            : base()
        {
            #region
            Initialization
            Credentials.ProjectName =
        }
    }
}
```

[Copy](#)

```
"LowExample";
    #endregion

    base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

    Notification.Comment($"Instrument {instrument.Symbol} low
price is
{instrument.DayInfo.Low}");
}
}
```

See Also

[Reference](#)

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoMatchVolume Property

Instrument's match volume value

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double MatchVolume {  
    get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
MatchVolumeExample :  
IndicatorBuilder  
    {  
        public  
MatchVolumeExample()  
            : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "MatchVolumeExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
}  
  
public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
    Notification.Comment($"Instrument {instrument.Symbol} match  
volume is  
{instrument.DayInfo.MatchVolume}");  
}
```

```
}
```

```
}
```

```
}
```

```
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoOpen Property

Instrument's last open price

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Open { get; }
```

[Copy](#)

Property Value
Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
OpenExample : IndicatorBuilder
    {
        public
OpenExample()
            : base()
        {
            #region
Initialization
        }
    }
}
```

[Copy](#)

```
Credentials.ProjectName =  
    "OpenExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
    public override  
void Init()  
{  
  
}  
  
    public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
Notification.Comment($"Instrument {instrument.Symbol} open  
price is  
{instrument.DayInfo.Open}");  
}  
}  
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoOpenInterest Property

Open interest price

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double OpenInterest {  
    get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
        OpenInterestExample :  
            IndicatorBuilder  
    {  
        public  
            OpenInterestExample()  
            : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "OpenInterestExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
}  
  
public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
    Notification.Comment($"Instrument {instrument.Symbol} open  
interest price is  
{instrument.DayInfo.OpenInterest}");  
}  
}  
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoPast SettlementPrice Property

Instrument's past settlement price value

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double  
PastSettlementPrice { get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
PastSettlementPriceExample :  
IndicatorBuilder  
{  
    public
```

[Copy](#)

```
PastSettlementPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"PastSettlementPriceExample";
    #endregion

base.SeparateWindow = false;
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

Notification.Comment($"Instrument {instrument.Symbol} past
settlement price is
{instrument.DayInfo.PastSettle
mentPrice}");
}
}
```

See Also

Reference

[DayInfo Class](#)

TradeApi.Instruments Namespace

DayInfoPrevClose Property

Instrument's previous close price

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double PrevClose { get; } Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;

namespace TestEnv
{
    public class
PrevCloseExample : IndicatorBuilder
    {
        public
PrevCloseExample()
            : base()
```

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "PrevCloseExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
}  
  
public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
    Notification.Comment($"Instrument {instrument.Symbol}  
previous close price is  
{instrument.DayInfo.PrevClose}  
");  
}  
}  
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoSettlement Price Property

Instrument's settlement price value

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double SettlementPrice
{ get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
SettlementPriceExample :  
IndicatorBuilder
{
    public
SettlementPriceExample()
    : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "SettlementPriceExample";  
    #endregion  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
}  
  
public override  
void Update(TickStatus args)  
{  
    var instrument =  
InstrumentsManager.Current;  
  
    Notification.Comment($"Instrument {instrument.Symbol}  
settlement price is  
{instrument.DayInfo.Settlement  
Price}");  
}  
}  
}
```

See Also

[Reference](#)

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoTicks Property

Instrument's ticks

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public long Ticks { get; }
```

[Copy](#)

Property Value

Int64

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
TicksExample :
IndicatorBuilder
{
    public
TicksExample()
        : base()
    {
        #region
Initialization
    }
}
```

[Copy](#)

```
Credentials.ProjectName =
"TicksExample";
#endregion

base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

Notification.Comment($"Instrument {instrument.Symbol} ticks
value is
{instrument.DayInfo.Ticks}");
}
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfoVolume Property

Instrument's volume value

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Volume { get; } Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
VolumeExample :
IndicatorBuilder
{
    public
VolumeExample()
    : base()
{
```

```
        #region
Initialization

Credentials.ProjectName =
"VolumeExample";
        #endregion

base.SeparateWindow = false;
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;

Notification.Comment($"Instrument {instrument.Symbol} match
volume is
{instrument.DayInfo.Volume}");
}
}
```

See Also

Reference

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DayInfo Methods

The [DayInfo](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[DayInfo Class](#)

[TradeApi.Instruments Namespace](#)

DepthAggregate Enumeration

Market depth aggregate type

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum DepthAggregate
```

[Copy](#)

► Members

Member name	Value	Description
ByPriceLevel	0	ByPriceLevel
ByOrder	1	ByOrder

► See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

DerivativeContract Class

DerivativeContract entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsDerivativeContract](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
DerivativeContract
```

[Copy](#)

The [DerivativeContract](#) type exposes the following members.

► Constructors

Name	Description
 DerivativeContract	Initializes a new instance of the DerivativeContract class

[Top](#)

Properties

	Name	Description
	ExpirationDate	Contract's expiry date
	Instrument	Contract's instrument
	Symbol	Contract's Symbol

[Top](#)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string

that represents
the current
object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

DerivativeContract Constructor

Initializes a new instance of the
[DerivativeContract](#) class

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public DerivativeContract()
```

[Copy](#)

◀ See Also

[Reference](#)

[DerivativeContract Class](#)

[TradeApi.Instruments Namespace](#)

DerivativeContract Properties

The [DerivativeContract](#) type exposes the following members.

Properties

	Name	Description
 	ExpirationDate	Contract's expiry date
 	Instrument	Contract's instrument
 	Symbol	Contract's Symbol

[Top](#)

See Also

[Reference](#)

[DerivativeContract Class](#)

[TradeApi.Instruments Namespace](#)

Derivative ContractExpiration Date Property

Contract's expiry date

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime ExpirationDate  
{ get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
ExpirationDateExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public  
ExpirationDateExample()  
        : base()  
{  
    #region  
Initialization  
  
Credentials.ProjectName =  
"ExpirationDateExample";  
    #endregion  
}  
  
Instrument  
instrument;  
  
        public override  
void Init()  
{  
    instrument =  
InstrumentsManager.Current;  
}  
  
        public override  
void Update(TickStatus args)  
{  
    if (args ==  
TickStatus.IsBar &&  
instrument.Type ==  
TradeApi.Instruments.Instrumen  
tType.Option)  
    {  
        var  
contracts =  
InstrumentsManager.GetDerivati  
veContracts(instrument);  
  
contracts.ForEach(x => {  
  
Notification.Print($"the  
contract's {x.Symbol} expiry
```

```
date:  
{x.ExpirationDate.ToString() }" );  
} );  
}  
}  
}
```

See Also

Reference

[DerivativeContract Class](#)

[TradeApi.Instruments Namespace](#)

Derivative ContractInstrument Property

Contract's instrument

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Instrument {  
    get; }
```

Copy

Property Value

Instrument

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
InstrumentExample :  
StrategyBuilder  
{
```

Copy

```
        public
InstrumentExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"InstrumentExample";
    #endregion
}

Instrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.Option)
    {
        var
contracts =
InstrumentsManager.GetDerivati
veContracts(instrument);

contracts.ForEach(x => {

Notification.Print($"the
contract's instrument symbol:
```

```
{x.Instrument.Symbol}");  
    } );  
}  
}  
}
```

See Also

[Reference](#)

[DerivativeContract Class](#)

[TradeApi.Instruments Namespace](#)

Derivative ContractSymbol Property

Contract's Symbol

Namespace: TradeApi.Instruments
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public string Symbol { get; } Copy
```

Property Value

String

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
SymbolExample :
StrategyBuilder
{
    public
```

```
SymbolExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"SymbolExample";
    #endregion
}

Instrument
instrument;

public override
void Init()
{
    instrument =
InstrumentsManager.Current;
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.Option)
    {
        var
contracts =
InstrumentsManager.GetDerivati
veContracts(instrument);

contracts.ForEach(x => {

Notification.Print($"the
contract's symbol:
{x.Symbol}");
}
```

```
        } ) ;  
    }  
}
```

◀ See Also

Reference

[DerivativeContract Class](#)

[TradeApi.Instruments Namespace](#)

DerivativeContract Methods

The [DerivativeContract](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[DerivativeContract Class](#)

[TradeApi.Instruments Namespace](#)

EquitiesInstrument Class

Equities instrument entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsEquitiesInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
EquitiesInstrument :  
Instrument
```

[Copy](#)

The [EquitiesInstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument

		(Inherited from Instrument)
	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotSize	Amount of base asset for one lot
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Symbol	Symbol of the base instrument

		(Inherited from Instrument)
 	TickSize	Array of minimal value size of pair's quote used by script
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

Methods

	Name	Description
 	Equals	Allow to compare two instruments (Inherited from Instrument)
 	GetHashCode	(Inherited from Instrument)
 	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[MaximalLot](#)

The biggest
trade allowed
(in lot)
(Inherited from
[Instrument](#))



[MinimalLot](#)

The smallest
trade allowed
(in lot)
(Inherited from
[Instrument](#))



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

EquitiesInstrument Properties

The [EquitiesInstrument](#) type exposes the following members.

Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotSize	Amount of base asset for one lot
 	LotStep	Step of the lot changes

		(Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TickSize	Array of minimal value size of pair's quote used by script
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

See Also

[Reference](#)

[EquitiesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

EquitiesInstrumentLot Size Property

Amount of base asset for one lot

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int LotSize { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
LotSizeExample :
StrategyBuilder
{
    public
LotSizeExample()
```

[Copy](#)

```
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"LotSizeExample";
        #endregion
    }
OrderRequest
request;

        public override
void Init()
{
        var instrument
= InstrumentsManager.Current;
        if(instrument
is EquitiesInstrument)
            request =
new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy,
instrument.MinimalLot(ProductT
ype.General) * (instrument as
EquitiesInstrument).LotSize);
    }

        public override
void Update(TickStatus args)
{
        if (args ==
TickStatus.IsBar && request !=
null)
    {

OrdersManager.Send(request);
    }
}
```

```
        }  
    }  
}
```



See Also

[Reference](#)

[EquitiesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Equities InstrumentTickSize Property

Array of minimal value size of pair's quote used by script

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public List<Ticksize> TickSize Copy
{ get; }
```

Property Value
[ListTicksize](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
    TickSizeExample :
    StrategyBuilder
```

```
        {
            public
    TickSizeExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"TickSizeExample";
            #endregion
        }
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Future)
    {
        var
tickSize =
(InstrumentsManager.Current as
EquitiesInstrument).TickSize;

tickSize.ForEach(element =>
Notification.Print($" tick
size: {element.TickSize};"));
    }
}
```

```
        }  
    }
```

See Also

Reference

[EquitiesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Equities InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value

[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
```

[Copy](#)

```
        public
TypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TypeExample";
    #endregion
}
OrderRequest
request;
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.Equity)
{
```

```
        OrdersManager.Send(request);  
    }  
}  
}  
}
```



▲ See Also

Reference

[EquitiesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

EquitiesInstrument Methods

The [EquitiesInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[EquitiesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ETFInstrument Class

ETF instrument entity

▪ Inheritance Hierarchy

SystemObject [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsETFInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
ETFInstrument : Instrument
```

[Copy](#)

The [ETFInstrument](#) type exposes the following members.

▪ Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)
 LimitHigh	Gets the high price limit to the

		selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)



Type

Market category
of the
instrument
(Overrides
[InstrumentType](#))

[Top](#)

Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot) (Inherited from Instrument)



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

See Also

Reference

[TradeApi.Instruments Namespace](#)

ETFIInstrument Properties

The [ETFIInstrument](#) type exposes the following members.

Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)

 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

See Also

[Reference](#)

[ETFInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ETFInstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value
[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
    public
TypeExample()
```

[Copy](#)

```
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"TypeExample";
        #endregion
    }
    OrderRequest
request;
    Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot());
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.ETF)
    {

OrdersManager.Send(request);
}
}
```

```
        }  
    }
```



See Also

Reference

[ETFIInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ETFInstrument Methods

The [ETFInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[ETFInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ForexInstrument Class

Forex instrument entity

► Inheritance Hierarchy

SystemObject [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsForexInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
ForexInstrument : Instrument
```

[Copy](#)

The [ForexInstrument](#) type exposes the following members.

► Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high

		price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotSize	Amount of base asset for one lot
 	LotStep	Step of the lot changes (Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Precision	Evaluates a precision unit of selected instrument
 	Symbol	Symbol of the

base instrument
(Inherited from
[Instrument](#))

 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

◀ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest

trade allowed
(in lot)
(Inherited from
[Instrument](#))



[MinimalLot](#)

The smallest
trade allowed
(in lot)
(Inherited from
[Instrument](#))



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

ForexInstrument Properties

The [ForexInstrument](#) type exposes the following members.

Properties

Name	Description
  DayInfo	Day bar information for an instrument (Inherited from Instrument)
  LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
  LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
  LotSize	Amount of base asset for one lot
  LotStep	Step of the lot changes

		(Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Precision	Evaluates a precision unit of selected instrument
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

See Also

[Reference](#)

[ForexInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ForexInstrumentLot Size Property

Amount of base asset for one lot

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int LotSize { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
LotSizeExample :
StrategyBuilder
{
    public
LotSizeExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"LotSizeExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
    if (instrument
is ForexInstrument &&
(instrument as
ForexInstrument).LotSize > 0)
        request =
new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy, (instrument as
ForexInstrument).LotSize);
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request != null)
{
```

```
        OrdersManager.Send(request);  
    }  
}  
}  
}
```



▲ See Also

Reference

[ForexInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Forex InstrumentPrecision Property

Evaluates a precision unit of selected instrument

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public int Precision { get; } Copy
```

Property Value

Int32

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;
using System;

namespace TestEnv
{
    public class
PrecisionExample :
```

```
StrategyBuilder
{
    public
PrecisionExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"PrecisionExample";
    #endregion
}
OrderRequest
request;
ForexInstrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current as
ForexInstrument;
    request = new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));

request.SLTPPriceType =
SLTPPriceType.Absolute;

request.StopLossPrice =
Math.Round(instrument.MinimalT
ickSize,
instrument.Precision);
}
```

```
    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Forex)
    {

OrdersManager.Send(request);
    }
}
}
```

◀ See Also

[Reference](#)

[ForexInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ForexInstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value
[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
    {
        public
TypeExample()
    }
}
```

[Copy](#)

```
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"TypeExample";
        #endregion
    }
    OrderRequest
request;
    Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Forex)
    {

OrdersManager.Send(request);
    }
}
```

```
    }  
}
```



See Also

Reference

[ForexInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ForexInstrument Methods

The [ForexInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[ForexInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ForwardInstrument Class

Forward instrument entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsForwardInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class ForwardInstrument : Instrument
```

[Copy](#)

The [ForwardInstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)

	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Symbol	Symbol of the base instrument (Inherited from Instrument)
	TradingStatus	Current trading status for the

instrument
(Inherited from
[Instrument](#))

 	Type	Market category of the instrument (Overrides InstrumentType)
---	----------------------	---

[Top](#)

◀ Methods

	Name	Description
 	Equals	Allow to compare two instruments (Inherited from Instrument)
 	GetHashCode	(Inherited from Instrument)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
 	MinimalLot	The smallest

trade allowed
(in lot)
(Inherited from
[Instrument](#))



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

ForwardInstrument Properties

The [ForwardInstrument](#) type exposes the following members.

Properties

Name	Description
  DayInfo	Day bar information for an instrument (Inherited from Instrument)
  LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
  LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
  LotStep	Step of the lot changes (Inherited from Instrument)

 MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 Symbol	Symbol of the base instrument (Inherited from Instrument)
 TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

See Also

[Reference](#)

[ForwardInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Forward InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value

[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
```

[Copy](#)

```
    public
TypeExample()
    : base()
{
    #region
Initialization

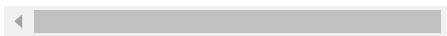
Credentials.ProjectName =
"TypeExample";
    #endregion
}
OrderRequest
request;
Instrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot());
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Forward)
    {

OrdersManager.Send(request);
    }
}
```

```
        }  
    }  
}
```



See Also

[Reference](#)

[ForwardInstrument Class](#)

[TradeApi.Instruments Namespace](#)

ForwardInstrument Methods

The [ForwardInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[ForwardInstrument Class](#)

[TradeApi.Instruments Namespace](#)

FuturesInstrument Class

Futures instrument entity

► Inheritance Hierarchy

SystemObject [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsFuturesInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
FuturesInstrument : Instrument
```

[Copy](#)

The [FuturesInstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)

 	ExpirationDate	The expiry date of the contract
 	FirstTradeDate	The date of first trade of the contract
 	LastTradeDate	The date of last trade of the contract
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotSize	Amount of base asset for one lot
 	LotStep	Step of the lot changes (Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)

 	MaturityDate	Contract month date
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TickSizeTickCost	Array of minimal value size of pair's quote used by script
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)
 	Underlayer	Underlayer Instrument

[Top](#)

◀ Methods

Name	Description
------	-------------

 Equals	Allow to compare two instruments (Inherited from Instrument)
 GetHashCode	(Inherited from Instrument)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
 MinimalLot	The smallest trade allowed (in lot) (Inherited from Instrument)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

Reference

[TradeApi.Instruments Namespace](#)

FuturesInstrument Properties

The [FuturesInstrument](#) type exposes the following members.

Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)
 ExpirationDate	The expiry date of the contract
 FirstTradeDate	The date of first trade of the contract
 LastTradeDate	The date of last trade of the contract
 LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 LimitLow	Gets the low

price limit to the selected instrument
(Inherited from [Instrument](#))

 	LotSize	Amount of base asset for one lot
 	LotStep	Step of the lot changes (Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MaturityDate	Contract month date
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TickSizeTickCost	Array of minimal value size of pair's quote used by script

	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)
 	Underlayer	Underlayer Instrument

[Top](#)

See Also

- [Reference](#)
- [FuturesInstrument Class](#)
- [TradeApi.Instruments Namespace](#)

Futures InstrumentExpiration Date Property

The expiry date of the contract

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime ExpirationDate
{ get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
ExpirationDateExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
ExpirationDateExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ExpirationDateExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
        if(instrument
is FuturesInstrument)
            request =
new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Buy,
instrument.MinimalLot(ProductT
ype.General));
    }

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request !=
```

```
    null)
    {
        var
        futures =
        InstrumentsManager.Current as
        FuturesInstrument;
        var
        ExpirationDate =
        futures.ExpirationDate >
        DateTime.UtcNow;

        if(ExpirationDate)

            OrdersManager.Send(request);
        }
    }
}
```

◀ See Also

Reference

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Futures InstrumentFirstTrade Date Property

The date of first trade of the contract

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime FirstTradeDate
{ get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;
using System;

namespace TestEnv
{
    public class
FirstTradeDateExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
FirstTradeDateExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"FirstTradeDateExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
        if(instrument
is FuturesInstrument)
            request =
new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Buy,
instrument.MinimalLot(ProductT
ype.General));
    }

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request !=
```

```
null)
{
    var
    futures =
    InstrumentsManager.Current as
    FuturesInstrument;

    if(futures.LastTradeDate <
    futures.FirstTradeDate)

    OrdersManager.Send(request);
}
}
}
```

◀ See Also

Reference

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Futures InstrumentLastTrade Date Property

The date of last trade of the contract

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public DateTime LastTradeDate
{ get; }
```

[Copy](#)

Property Value
[DateTime](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;
using System;

namespace TestEnv
{
    public class
LastTradeDateExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
LastTradeDateExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"LastTradeDateExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
        if(instrument
is FuturesInstrument)
            request =
new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Buy,
instrument.MinimalLot(ProductT
ype.General));
    }

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request !=
```

```
null)
{
    var
futures =
InstrumentsManager.Current as
FuturesInstrument;

if(futures.LastTradeDate <
DateTime.UtcNow)

OrdersManager.Send(request);
}
}
}
```

◀ See Also

Reference

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

FuturesInstrumentLot Size Property

Amount of base asset for one lot

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int? LotSize { get; } Copy
```

Property Value

[NullableInt32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
LotSizeExample :
StrategyBuilder
{
    public
LotSizeExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "LotSizeExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
    if (instrument
is FuturesInstrument)
        request =
new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request !=
null)
    {

OrdersManager.Send(request);
```

```
        }  
    }  
}
```



◀ See Also

Reference

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Futures InstrumentMaturity Date Property

Contract month date

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime MaturityDate {  
    get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi.Instruments;  
using System;  
  
namespace TestEnv  
{  
    public class  
MaturityDateExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
MaturityDateExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"MaturityDateExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
        if(instrument
is FuturesInstrument)
            request =
new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy, (instrument as
FuturesInstrument).LotSize ??
instrument.MinimalLot(ProductT
ype.General));
    }

        public override
void Update(TickStatus args)
{
    if (args ==

```

```
    TickStatus.IsBar && request !=  
    null)  
    {  
        var  
        futures =  
        InstrumentsManager.Current as  
        FuturesInstrument;  
        var  
        isMatured =  
        futures.MaturityDate <  
        DateTime.UtcNow;  
  
        if (!isMatured)  
  
            OrdersManager.Send(request);  
        }  
    }  
}
```



See Also

Reference

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Futures InstrumentTickSize TickCost Property

Array of minimal value size of pair's quote used by script

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public List<TicksizeTickcost>
    TickSizeTickCost { get; }
```

[Copy](#)

Property Value
[ListTicksizetickcost](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
        TickSizeTickCostExample :
            StrategyBuilder
```

[Copy](#)

```
        {
            public
TickSizeTickCostExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"TickSizeTickCostExample";
    #endregion
}
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Future)
    {
        var
tickSize =
(InstrumentsManager.Current as
FuturesInstrument).TickSizeTic
kCost;

tickSize.ForEach(element =>
Notification.Print($" tick
size: {element.TickSize}; tick
cost: {element.TickCost}"));
}
```

```
        }  
    }  
}
```

See Also

Reference

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Futures InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value

[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
```

[Copy](#)

```
    public
TypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TypeExample";
    #endregion
}
OrderRequest
request;
Instrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Future)
{

OrdersManager.Send(request);
```

```
        }  
    }  
}
```



◀ See Also

Reference

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Futures InstrumentUnderlayer Property

Underlayer Instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Underlayer {  
    get; }
```

[Copy](#)

Property Value

[Instrument](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
UnderlayerExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public
UnderlayerExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"UnderlayerExample";
    #endregion
}
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Future)
    {
        var
underlayer =
(InstrumentsManager.Current as
FuturesInstrument).Underlayer?
.Symbol;

if(underlayer != string.Empty)

Notification.Print($" the
futures instrument:
{InstrumentsManager.Current.Sy
```

```
mbol} has underlaying asset:  
{underlayer}" );  
}  
}  
}
```

See Also

[Reference](#)

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

FuturesInstrument Methods

The [FuturesInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[FuturesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

IndicesInstrument Class

Indices instrument entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsIndicesInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class IndicesInstrument : Instrument
```

[Copy](#)

The [IndicesInstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)

	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Symbol	Symbol of the base instrument (Inherited from Instrument)
	TradingStatus	Current trading status for the

instrument
(Inherited from
[Instrument](#))

 	Type	Market category of the instrument (Overrides InstrumentType)
---	----------------------	---

[Top](#)

◀ Methods

	Name	Description
 	Equals	Allow to compare two instruments (Inherited from Instrument)
 	GetHashCode	(Inherited from Instrument)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
 	MinimalLot	The smallest

trade allowed
(in lot)
(Inherited from
[Instrument](#))



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

IndicesInstrument Properties

The [IndicesInstrument](#) type exposes the following members.

Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)

 MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 Symbol	Symbol of the base instrument (Inherited from Instrument)
 TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

See Also

[Reference](#)

[IndicesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Indices InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value

[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
```

[Copy](#)

```
    public
TypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TypeExample";
    #endregion
}
OrderRequest
request;
Instrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot());
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Indice)
    {

OrdersManager.Send(request);
    }
}
```

```
        }  
    }  
}
```

◀ ▶

See Also

[Reference](#)

[IndicesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

IndicesInstrument Methods

The [IndicesInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[IndicesInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Instrument Class

Instrument entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsInstrument](#)

[More](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public abstract class  
Instrument : IConvertible,  
IDisposable
```

[Copy](#)

The [Instrument](#) type exposes the following members.

► Properties

	Name	Description
	DayInfo	Day bar information for an instrument
	LimitHigh	Gets the high price limit to

		the selected instrument
 	LimitLow	Gets the low price limit to the selected instrument
 	LotStep	Step of the lot changes
 	MarketDepth	MarketDepth info for an instrument
 	MinimalTickSize	Min value size of pair's quote used by script
 	Symbol	Symbol of the base instrument
 	TradingStatus	Current trading status for the instrument
 	Type	Market category of the instrument

[Top](#)

Methods

Name	Description
	

Equals

Allow to compare two instruments
(Overrides
[Object.Equals\(Object\)](#))



Finalize

(Overrides
[Object.Finalize](#))



GetHashCode

(Overrides
[Object.GetHashCode](#))



GetType

Gets the [Type](#) of the current instance.
(Inherited from
[Object](#))



MaximalLot

The biggest trade allowed (in lot)



MemberwiseClone

Creates a shallow copy of the current [Object](#).
(Inherited from
[Object](#))



MinimalLot

The smallest trade allowed (in lot)



ToString

Returns a string that represents the current object.
(Inherited from
[Object](#))

[Top](#)

Operators

	Name	Description
 S	Equality(Instrument, Instrument)	Allow to compare two instruments
 S	Inequality(Instrument, Instrument)	Allow to compare two instruments

[Top](#)

↳ Fields

	Name	Description
	_internalInstrument	

[Top](#)

↳ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

↳ Inheritance Hierarchy

[SystemObject](#)

[TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsBondsInstrument](#)

[TradeApi.InstrumentsCFDInstrument](#)

[TradeApi.InstrumentsCorporateInstrument](#)

[TradeApi.InstrumentsCryptoInstrument](#)

[TradeApi.InstrumentsEquitiesInstrument](#)

[TradeApi.InstrumentsETFIInstrument](#)

[TradeApi.InstrumentsForexInstrument](#)

[TradeApi.InstrumentsForwardInstrument](#)

TradeApi.InstrumentsFuturesInstrument
TradeApi.InstrumentsIndicesInstrument
TradeApi.InstrumentsOptionsInstrument
TradeApi.InstrumentsSpotInstrument
TradeApi.InstrumentsSpreadBetInstrume
nt
TradeApi.InstrumentsSyntheticInstrumen
t
TradeApi.InstrumentsTBillsIsnstrument

Instrument Properties

The [Instrument](#) type exposes the following members.

Properties

	Name	Description
	DayInfo	Day bar information for an instrument
	LimitHigh	Gets the high price limit to the selected instrument
	LimitLow	Gets the low price limit to the selected instrument
	LotStep	Step of the lot changes
	MarketDepth	MarketDepth info for an instrument
	MinimalTickSize	Min value size of pair's quote used by script

	Symbol	Symbol of the base instrument
	TradingStatus	Current trading status for the instrument
	Type	Market category of the instrument

[Top](#)

▲ See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentDayInfo Property

Day bar information for an instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DayInfo DayInfo { get; }Copy
```

Property Value

[DayInfo](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
DayInfoExample :
StrategyBuilder
{
    public
DayInfoExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "DayInfoExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
instrument =
InstrumentsManager.Current;

Notification.Print($"Instrument's daily volume:
{instrument.DayInfo.Volume}");
    }
}
}
```

See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentLimitHigh Property

Gets the high price limit to the selected instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public double LimitHigh { get; } Copy
```

Property Value

[Double](#)

▪ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
LimitHighExample :
StrategyBuilder
{
    public
```

```
    LimitHighExample()
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"LimitHighExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
= InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{

if (InstrumentsManager.Current.
LimitHigh >
InstrumentsManager.Current.Day
Info.Ask)
{
    var
```

```
instrument =
InstrumentsManager.Current;

request.SLTPPriceType =
SLTPPriceType.Absolute;

request.Price =
instrument.DayInfo.Ask +
instrument.MinimalTickSize *
50;

OrdersManager.Send(request);
}
}
}
}
```



See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentLimitLow Property

Gets the low price limit to the selected instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public double LimitLow { get; } Copy
```

Property Value

[Double](#)

▪ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
LimitLowExample :
StrategyBuilder
{
    public
```

```
    LimitLowExample()
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"LimitLowExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
= InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{

if (InstrumentsManager.Current.
LimitLow <
InstrumentsManager.Current.Day
Info.Bid)
{
    var
```

```
instrument =
InstrumentsManager.Current;

request.SLTPPriceType =
SLTPPriceType.Absolute;

request.Price =
instrument.DayInfo.Bid -
instrument.MinimalTickSize *
50;

OrdersManager.Send(request);
}
}
}
}
```

◀ See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentLotStep Property

Step of the lot changes

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double LotStep { get; } Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
LotStepExample :
StrategyBuilder
{
    public
LotStepExample()
    : base()
```

```
        {
            #region
Initialization

Credentials.ProjectName =
"LotStepExample";
            #endregion
        }
OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
    var quantity =
2 * instrument.LotStep *
instrument.MinimalLot(ProductT
ype.General);
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy, quantity);
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
instrument =
InstrumentsManager.Current;

request.SLTPPriceType =

```

```
SLTPPriceType.Absolute;

request.Price =
instrument.DayInfo.Ask +
instrument.MinimalTickSize *
50;

OrdersManager.Send(request);
}
}
}
```



See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentMarketDepth Property

MarketDepth info for an instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public MarketDepth MarketDepth
{ get; }
```

[Copy](#)

Property Value

[MarketDepth](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
MarketDepthExample : 
StrategyBuilder
{
    public
MarketDepthExample()
        : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"MarketDepthExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
InstrumentsManager.Current.Type ==
TradeApi.Instruments.InstrumentType.Future)
    {
        var
instrument =
InstrumentsManager.Current;
        var quotes
=
instrument.MarketDepth.DepthAsk(TradeApi.Instruments.DepthAggregate.ByOrder);

quotes.ForEach(level2 =>
Notification.Print($""
{level2.Price}"));
    }
}
```

```
    }  
}
```



See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentMinimal TickSize Property

Min value size of pair's quote used by script

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double MinimalTickSize
{ get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
MinimalTickSizeExample :
StrategyBuilder
{
    public
MinimalTickSizeExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"MinimalTickSizeExample";
        #endregion
    }

        OrderRequest
request;

        public override
void Init()
{
    var instrument
= InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
tickSize =
InstrumentsManager.Current.Min
imalTickSize;
        var
instrument =
InstrumentsManager.Current;
```

```
request.SLTPPriceType =  
SLTPPriceType.Absolute;  
  
request.Price =  
instrument.DayInfo.Bid -  
tickSize * 50;  
  
request.StopLossPrice =  
tickSize * 20 +  
InstrumentsManager.Current.Day  
Info.Ask;  
  
OrdersManager.Send(request);  
    }  
}  
}  
}
```



See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentSymbol Property

Symbol of the base instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Symbol { get; } Copy
```

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
SymbolExample :
StrategyBuilder
{
    public
SymbolExample()
    : base()
```

```
        {
            #region
            Initialization

            Credentials.ProjectName =
"SymbolExample";
            #endregion
        }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
position =>
position.Instrument.Symbol ==
InstrumentsManager.Current.Sym
bol);

Notification.Print($"Position'
s count: {positions.Count}");
    }
}
}
```

See Also

Reference

[Instrument Class](#)

TradeApi.Instruments Namespace

InstrumentTrading Status Property

Current trading status for the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TradingStatus  
TradingStatus { get; }
```

[Copy](#)

Property Value

[TradingStatus](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
TradingStatusExample :  
StrategyBuilder  
    {  
        public  
TradingStatusExample()
```

[Copy](#)

```
        : base()
    {
        #region
Initialization

        Credentials.ProjectName =
"TradingStatusExample";
        #endregion
    }
OrderRequest
request;

        public override
void Init()
{
        var instrument
= InstrumentsManager.Current;
        request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
        if (args ==
TickStatus.IsBar)
{

if (InstrumentsManager.Current.
TradingStatus ==
TradeApi.Instruments.TradingSt
atus.Open)
{

var instrument =
```

```
InstrumentsManager.Current;

request.SLTPPriceType =
SLTPPriceType.Absolute;

request.Price =
instrument.DayInfo.Ask +
instrument.MinimalTickSize *
50;

OrdersManager.Send(request);
}

}

}

}

}
```

See Also

Reference

Instrument Class

TradeApi.Instruments Namespace

InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public abstract InstrumentType  
Type { get; }
```

[Copy](#)

Property Value
[InstrumentType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
TypeExample : StrategyBuilder  
    {  
        public  
TypeExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"TypeExample";
        #endregion
    }
    OrderRequest
request;
    Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Future)
    {
        var
instrument =
InstrumentsManager.Current;
```

```
    request.SLTPPriceType =  
    SLTPPriceType.Absolute;  
  
    request.Price =  
    instrument.DayInfo.Bid -  
    instrument.MinimalTickSize *  
    50;  
  
    OrdersManager.Send(request);  
}  
}  
}  
}
```



See Also

Reference

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

Instrument Methods

The [Instrument](#) type exposes the following members.

Methods

	Name	Description
	Equals	Allow to compare two instruments (Overrides Object.Equals(Object))
	Finalize	(Overrides Object.Finalize)
	GetHashCode	(Overrides Object.GetHashCode)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot)
	MemberwiseClone	Creates a shallow copy of the current Object . (Inherited from Object)

[MinimalLot](#)

The smallest trade allowed (in lot)



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentEquals Method

Allow to compare two instruments

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override bool Equals(Copy
                           Object obj
                         )
```

Parameters

obj [Object](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentFinalize Method

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
protected override void  
Finalize()
```

[Copy](#)

Implements
[ObjectFinalize](#)

◀ See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentGetHash Code Method

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override int  
GetHashCode()
```

[Copy](#)

Return Value

[Int32](#)

► See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentMaximalLot Method

The biggest trade allowed (in lot)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double MaximalLot(Copy
    ProductType  
productType =  
ProductType.General  
)
```

Parameters

productType [ProductType](#) (Optional)
specified product type

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;Copy  
using TradeApi.Trading;  
using TradeApi.History;
```

```
namespace TestEnv
{
    public class MaximalLotExample : StrategyBuilder
    {
        public MaximalLotExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "MaximalLotExample";
            #endregion
        }
        OrderRequest request;
        public override void Init()
        {
            var instrument =
                HistoryDataSeries.HistoricalRequest.Instrument;
            request = new OrderRequest(OrderType.Limit,
                instrument,
                AccountManager.Current,
                OrderSide.Buy,
                instrument.MaximalLot(ProductType.General));
        }

        public override void Update(TickStatus args)
        {
            if (args ==

```

```
    TickStatus.IsBar)
    {
        OrdersManager.Send(request);
    }
}
```



See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentMinimalLot Method

The smallest trade allowed (in lot)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double MinimalLot(Copy
    ProductType  
productType =  
ProductType.General  
)
```

Parameters

productType [ProductType](#) (Optional)
specified product type

Return Value

[Double](#)

► Example

C#

```
using Runtime.Script;Copy  
using TradeApi.Trading;  
using TradeApi.History;
```

```
namespace TestEnv
{
    public class MinimalLotExample : StrategyBuilder
    {
        public MinimalLotExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "MinimalLotExample";
            #endregion
        }
        OrderRequest
        request;
    }

    public override
    void Init()
    {
        var instrument
        =
        HistoryDataSeries.HistoricalRe
        quest.Instrument;
        request = new
        OrderRequest(OrderType.Limit,
        instrument,
        AccountManager.Current,
        OrderSide.Buy,
        instrument.MinimalLot(ProductT
        ype.General));
    }

    public override
    void Update(TickStatus args)
    {
        if (args ==

```

```
    TickStatus.IsBar)
        {
            OrdersManager.Send(request);
        }
    }
}
```



See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

Instrument Operators

The [Instrument](#) type exposes the following members.

Operators

Name	Description
 S Equality(Instrument, Instrument)	Allow to compare two instruments
 S Inequality(Instrument, Instrument)	Allow to compare two instruments

[Top](#)

See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentEquality Operator

Allow to compare two instruments

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator ==  
(  
    Instrument instr1,  
    Instrument instr2  
)
```

[Copy](#)

Parameters

instr1 [Instrument](#)
instr2 [Instrument](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentInequality Operator

Allow to compare two instruments

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator !=  
(  
    Instrument instr1,  
    Instrument instr2  
)
```

[Copy](#)

Parameters

instr1 [Instrument](#)
instr2 [Instrument](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

Instrument Fields

The [Instrument](#) type exposes the following members.

► Fields

Name	Description
 _internalInstrument	

[Top](#)

► See Also

[Reference](#)

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

Instrument_Internal Instrument Field

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
protected readonly  
[T:5zA=.8Ng=]  
_internalInstrument
```

[Copy](#)

Field Value

[T:5zA=.8Ng=]

▪ See Also

Reference

[Instrument Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentsManager Class

InstrumentsManager entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.Instruments](#)
InstrumentsManager

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public sealed class InstrumentsManager
```

[Copy](#)

The InstrumentsManager type exposes the following members.

▪ Properties

	Name	Description
 	Current	Retrieves current instrument

[Top](#)

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)

 	GetDerivativeContacts	Gets option's instruments collection
 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetInstruments(DerivativeContract)	Gets option's instruments collection
 	GetInstruments(String)	Retrieves instrument by name
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	Subscribe	The Subscribe method is used to create a subscription to a resource. This method is used to specify the details about the event to be monitored: what instrument use to monitored;

what the type of quotes uses will be subscribed.

	ToString	Returns a string that represents the current object. (Inherited from Object)
---	--------------------------	--

 	UnSubscribe	The Unsubscribe operation is used to end a pull subscription. The Unsubscribe element contains the parameters used to unsubscribe from a subscription.
---	-----------------------------	--

[Top](#)

Events

Name	Description
 	OnBBO Occurs when the new best bid offer quote by the subscribed instrument is coming
 	OnInstrumentAdded Occurs when the new instrument is

coming



[OnLevel2](#)

Occurs when the new level2 quote by the subscribed instrument is coming



[OnTrades](#)

Occurs when the new trade quote by the subscribed instrument is coming

[Top](#)

See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

InstrumentsManager Properties

The [InstrumentsManager](#) type exposes the following members.

► Properties

	Name	Description
 	Current	Retrieves current instrument

[Top](#)

► See Also

[Reference](#)

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerCurrent Property

Retrieves current instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Current {  
    get; }
```

[Copy](#)

Property Value

[Instrument](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
Currentexample :  
StrategyBuilder  
{
```

```
        public
Currentxample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"Currentxample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
positions =
PositionsManager.GetPositions(
position =>
position.Instrument.Symbol ==
InstrumentsManager.Current.Sym
bol);

Notification.Print($"Position'
s count: {positions.Count}");
}
}
}
```

See Also

Reference

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentsManager Methods

The [InstrumentsManager](#) type exposes the following members.

▲ Methods

Name	Description
  Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
  GetDerivativeContacts	Gets option's instruments collection
  GetHashCode	Serves as the default hash function. (Inherited from Object)
  GetInstruments(DerivativeContract)	Gets option's instruments collection
  GetInstruments(String)	Retrieves instrument by name
  GetType	Gets the Type of the

current
instance.
(Inherited
from [Object](#))



[Subscribe](#)

The
Subscribe
method is
used to
create a
subscription
to a
resource. This
method is
used to
specify the
details about
the event to
be
monitored:
what
instrument
use to
monitored;
what the
type of
quotes uses
will be
subscribed.



[ToString](#)

Returns a
string that
represents
the current
object.
(Inherited
from [Object](#))



[UnSubscribe](#)

The
Unsubscribe
operation is
used to end
a pull
subscription.
The

Unsubscribe
element
contains the
parameters
used to
unsubscribe
from a
subscription.

[Top](#)

See Also

[Reference](#)

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerGetDerivative Contacts Method

Gets option's instruments collection

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public  
List<DerivativeContract>  
GetDerivativeContacts (  
    Instrument instrument  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)
derivative contract seria

Return Value

[ListDerivativeContract](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;
```

[Copy](#)

```
using TradeApi.History;
using TradeApi.Instruments;
using System;
using System.Linq;

namespace TestEnv
{
    public class
GetDerivativeContactsExample : 
StrategyBuilder
{
    public
GetDerivativeContactsExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetDerivativeContactsExample"
;
        #endregion
    }

    public Interval
interval;
    public BBOData
tickData;

    public override
void Init()
{
    var
derivativeContacts =
InstrumentsManager.GetDerivativeContacts(InstrumentsManager.
Current);
    if
(derivativeContacts!=null &&
derivativeContacts.Count > 0)
```

```
        {
            var
derivativeContract =
derivativeContacts.First();
            var
instrumentList =
InstrumentsManager.GetInstrume
nts(derivativeContract);

if(instrumentList!=null &&
instrumentList.Count > 0)
{
            var
tickRequest = new
TickHistoricalRequest(instrume
ntList.First(), DataType.Bid,
20);

interval = new
Interval(DateTime.UtcNow.AddDays(-2), DateTime.UtcNow);

tickData =
HistoricalDataManager.Get(tick
Request, interval) as BBOData;
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

InstrumentsManager Class
TradeApi.Instruments Namespace

InstrumentsManagerGet Instruments Method

▪ Overload List

Name	Description
  GetInstruments(DerivativeContract)	Gets option's instruments collection
  GetInstruments(String)	Retrieves instrument by name

[Top](#)

▪ See Also

[Reference](#)

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerGet Instruments(Derivative Contract) Method

Gets option's instruments collection

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public List<Instrument>
GetInstruments(
    DerivativeContract
seria
)
```

[Copy](#)

Parameters

seria [DerivativeContract](#)
derivative contract seria

Return Value

[ListInstrument](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;
using System;
using System.Linq;

namespace TestEnv
{
    public class
GetInstrumentsExample : StrategyBuilder
    {
        public
GetInstrumentsExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"GetInstrumentsExample";
            #endregion
        }

        public Interval
interval;
        public BBOData
tickData;

        public override
void Init()
        {
            var
derivativeContacts =
InstrumentsManager.GetDerivativeContacts(InstrumentsManager.
Current);
            if
(derivativeContacts!=null &&
```

```
derivativeContacts.Count > 0)
{
    var
derivativeContract =
derivativeContacts.First();
    var
instrumentList =
InstrumentsManager.GetInstrume
nts(derivativeContract);

if(instrumentList!=null &&
instrumentList.Count > 0)
{
    var
tickRequest = new
TickHistoricalRequest(instrume
ntList.First(), DataType.Bid,
20);

interval = new
Interval(DateTime.UtcNow.AddDays(-2), DateTime.UtcNow);

tickData =
HistoricalDataManager.Get(tick
Request, interval) as BBOData;
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[InstrumentsManager Class](#)

[GetInstruments Overload](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerGet Instruments(String) Method

Retrieves instrument by name

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public List<Instrument>
    GetInstruments(
        string symbol
    )
```

[Copy](#)

Parameters

symbol [String](#)
symbol of the instrument

Return Value
[ListInstrument](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using System.Linq;
using System;

namespace TestEnv
{
    public class
GetInstrumentExample : 
StrategyBuilder
{
    public
GetInstrumentExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetInstrumentExample";
        #endregion
    }

    public Interval
interval;

    public override void
Init()
    {
        var tickRequest =
new
TickHistoricalRequest(InstrumentsM
anager.GetInstruments("BRENT").Fir
stOrDefault(), DataType.Bid, 20);
        interval = new
Interval(DateTime.UtcNow.AddDays(-
2), DateTime.UtcNow);
        var tickData =
HistoricalDataManager.Get(tickRequ
```

```
    est, interval) as BBOData;
}

        public override void
Update(TickStatus args)
{
}

}

}
```

See Also

Reference

[InstrumentsManager Class](#)

[GetInstruments Overload](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerSubscribe Method

The Subscribe method is used to create a subscription to a resource. This method is used to specify the details about the event to be monitored: what instrument use to monitored; what the type of quotes uses will be subscribed.

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool Subscribe(Copy
    Instrument instrument,
    QuoteTypes types
)
```

Parameters

instrument [Instrument](#)

instrument to be subscribed to

types [QuoteTypes](#)

quote type to be subscribed to

Return Value

[Boolean](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;
using System;
using System.Linq;

namespace TestEnv
{
    public class
SubscribeExample : 
StrategyBuilder
{
    public
SubscribeExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SubscribeExample";
        #endregion
    }

    public override
void Init()
    {

InstrumentsManager.Subscribe(I
nstrumentsManager.Current,
TradeApi.Quotes.QuoteTypes.Quote);

InstrumentsManager.OnBBO +=
(bbo) => {
```

Copy

```
Notification.Comment($"Instrument's {bbo.Instrument.Symbol}  
ask is {bbo.Ask}");  
    };  
}  
  
    public override  
void Update(TickStatus args)  
{  
  
}  
  
    public override  
void Complete()  
{  
  
InstrumentsManager.UnSubscribe  
(InstrumentsManager.Current,  
TradeApi.Quotes.QuoteTypes.Quote);  
    }  
}  
}
```

See Also

Reference

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerUnSubscribe Method

The Unsubscribe operation is used to end a pull subscription. The Unsubscribe element contains the parameters used to unsubscribe from a subscription.

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool UnSubscribe(  
    Instrument instrument,  
    QuoteTypes types  
)
```

[Copy](#)

Parameters

instrument [Instrument](#)

instrument to be subscribed to

types [QuoteTypes](#)

quote type to be subscribed to

Return Value

[Boolean](#)

► Example

C#[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using
TradeApi.Instruments;

namespace TestEnv
{
    public class
UnSubscribeExample : StrategyBuilder
    {
        public
UnSubscribeExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"UnSubscribeExample";
            #endregion
        }

        public override
void Init()
        {

InstrumentsManager.Subscribe(InstrumentsManager.Current,
TradeApi.Quotes.QuoteTypes
.Quote);

InstrumentsManager.OnBBO
+= (bbo) => {

Notification.Comment($"Ins
trument's
```

```
    { bbo.Instrument.Symbol }
    ask is {bbo.Ask} );
}
}

public override
void Update(TickStatus
args)
{
}

public override
void Complete()
{
    InstrumentsManager.UnSubsc
    rive(InstrumentsManager.Cu
    rrent,
    TradeApi.Quotes.QuoteTypes
    .Quote);
}
}
```

See Also

Reference

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentsManager Events

The [InstrumentsManager](#) type exposes the following members.

Events

Name	Description
 OnBBO	Occurs when the new best bid offer quote by the subscribed instrument is coming
 OnInstrumentAdded	Occurs when the new instrument is coming
 OnLevel2	Occurs when the new level2 quote by the subscribed instrument is coming
 OnTrades	Occurs when the new trade quote

by the
subscribed
instrument
is coming

[Top](#)

◀ See Also

[Reference](#)

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerOnBBO Event

Occurs when the new best bid offer quote by the subscribed instrument is coming

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event Action<BBO> OnBBO Copy
```

Value

[ActionBBO](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
OnBBOExample : StrategyBuilder
    {
        public
OnBBOExample()
            : base()
        {
```

```
        #region
Initialization

Credentials.ProjectName =
"OnBBOExample";
        #endregion
    }

        public override
void Init()
{
    InstrumentsManager.OnBBO +=
(bbo) => {

Notification.Comment($"Instrument's {bbo.Instrument.Symbol} ask is {bbo.Ask}");
    };
}

        public override
void Update(TickStatus args)
{
    }
}
```

See Also

[Reference](#)

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

InstrumentsManagerOnInstrumentAdded Event

Occurs when the new instrument is coming

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event  
Action<Instrument>  
OnInstrumentAdded
```

[Copy](#)

Value

[ActionInstrument](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OnInstrumentAddedExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public
    OnInstrumentAddedExample()
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "OnInstrumentAddedExample";
        #endregion
    }

        public override
void Init()
{
}

InstrumentsManager.OnInstrumentAdded += (inst) => {

    Notification.Comment($"New
instrument {inst.Symbol} is
attached");
}

        public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerOnLevel2 Event

Occurs when the new level2 quote by the subscribed instrument is coming

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public event Action<Level2>
OnLevel2
```

[Copy](#)

Value
[ActionLevel2](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
OnLevel2Example :
StrategyBuilder
{
```

[Copy](#)

```
        public  
        OnLevel2Example()  
            : base()  
        {  
            #region  
            Initialization  
  
            Credentials.ProjectName =  
            "OnLevel2Example";  
            #endregion  
        }  
  
        public override  
        void Init()  
        {  
  
            InstrumentsManager.OnLevel2 +=  
            (level2) => {  
  
                Notification.Comment($"Exchange's name is {level2.MPID}");  
            };  
        }  
  
        public override  
        void Update(TickStatus args)  
        {  
  
        }  
    }  
}
```

See Also

Reference

[InstrumentsManager Class](#)

[TradeApi.Instruments Namespace](#)

Instruments ManagerOnTrades Event

Occurs when the new trade quote by the subscribed instrument is coming

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public event Action<Trade>  
OnTrades
```

[Copy](#)

Value
[ActionTrade](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OnTradesExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public
OnTradesExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"OnTradesExample";
    #endregion
}

        public override
void Init()
{
    InstrumentsManager.OnTrades +=
(trade) => {

if(trade.Aggressive >
TradeApi.Quotes.AggressiveFlag.
Ask)

Notification.Comment($"Instrum
ent {trade.Instrument.Symbol}
is bought");
    };
}

        public override
void Update(TickStatus args)
{
    }
}
```

See Also

Reference

InstrumentsManager Class

TradeApi.Instruments Namespace

InstrumentType Enumeration

Supported instrument type

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum InstrumentType
```

[Copy](#)

► Members

Member name	Value	Description
Crypto	0	Crypto
Equity	1	Equity
Forex	2	Forex
Future	3	Future
Option	4	Option
CFD	5	Synthetic
Forward	6	Forward
Indice	7	Indice

SpreadBet	8	SpreadBet
Bond	9	Bond
ETF	10	ETF
TBill	11	TBill
Spot	12	Spot
Corporate	13	Corporate
Synthetic	14	Synthetic

See Also

Reference

[TradeApi.Instruments Namespace](#)

MarketDepth Class

Instrument's market depth entity

▪ Inheritance Hierarchy

[SystemObject](#) `TradeApi.InstrumentsMarketDepth`

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
MarketDepth : IDisposable
```

[Copy](#)

The `MarketDepth` type exposes the following members.

▪ Methods

	Name	Description
	DepthAsk	Show depth of market by ask price
	DepthBid	Show depth of market by bid price
	Equals	Determines

whether the specified object is equal to the current object.
(Inherited from [Object](#))



[Finalize](#)

(Overrides [ObjectFinalize](#))



[GetHashCode](#)

Serves as the default hash function.
(Inherited from [Object](#))



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

▲ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

MarketDepth Methods

The [MarketDepth](#) type exposes the following members.

▪ Methods

	Name	Description
	DepthAsk	Show depth of market by ask price
	DepthBid	Show depth of market by bid price
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	Finalize	(Overrides ObjectFinalize)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type

of the current instance.
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[MarketDepth Class](#)

[TradeApi.Instruments Namespace](#)

MarketDepthDepthAsk Method

Show depth of market by ask price

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Level2> DepthAsk(Copy
    DepthAggregate
    aggregate =
    DepthAggregate.ByPriceLevel
)
```

Parameters

aggregate [DepthAggregate](#) (Optional)
depth aggregation type

Return Value

[ListLevel2](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Indicators;

namespace TestEnv
```

```
    {
        public class
DepthAskExample :
IndicatorBuilder
{
    public
DepthAskExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"DepthAskExample";
    #endregion

base.SeparateWindow = false;
}

public override
void Init()
{

}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;
    var depthList =
instrument.MarketDepth.DepthAs
k(TradeApi.Instruments.DepthAg
gregate.ByOrder);

depthList.ForEach(depth =>
Notification.Comment($"Exchang
e {depth.MPID} current price
is {depth.Price}"));
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[MarketDepth Class](#)

[TradeApi.Instruments Namespace](#)

MarketDepthDepthBid Method

Show depth of market by bid price

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Level2> DepthBid(Copy
    DepthAggregate
    aggregate =
    DepthAggregate.ByPriceLevel
)
```

Parameters

aggregate [DepthAggregate](#) (Optional)
depth aggregation type

Return Value

[ListLevel2](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Indicators;

namespace TestEnv
```

```
    {
        public class
DepthBidExample :
IndicatorBuilder
{
    public
DepthBidExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"DepthBidExample";
    #endregion

base.SeparateWindow = false;
}

public override
void Init()
{

}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;
    var depthList =
instrument.MarketDepth.DepthBi
d(TradeApi.Instruments.DepthAg
gregate.ByOrder);

depthList.ForEach(depth =>
Notification.Comment($"Exchang
e {depth.MPID} current price
is {depth.Price}"));
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[MarketDepth Class](#)

[TradeApi.Instruments Namespace](#)

MarketDepthFinalize Method

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
protected override void  
Finalize()
```

[Copy](#)

Implements
[ObjectFinalize](#)

◀ See Also

[Reference](#)

[MarketDepth Class](#)

[TradeApi.Instruments Namespace](#)

OptionsInstrument Class

Options instrument entity

► Inheritance Hierarchy

SystemObject [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsOptionsInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class OptionsInstrument : Instrument
```

[Copy](#)

The [OptionsInstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)

 	ExpirationDateUtc	The UTC expiry date of the contract
 	FirstTradeDateUtc	The UTC date of first trade of the contract
 	LastTradeDateUtc	The UTC date of last trade of the contract
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotSize	Amount of base asset for one lot
 	LotStep	Step of the lot changes (Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)

 	MaturityDateUtc	Contract month date in UTC
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	OptionType	Put/Call price type
 	StrikePrice	Strike prices which established when a contract is first written
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TickSizeTickCost	Array of minimal value size of pair's quote used by script
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument

(Overrides
[InstrumentType](#))



[Underlayer](#)

Underlayer
Instrument

[Top](#)

◀ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot) (Inherited from Instrument)



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

OptionsInstrument Properties

The [OptionsInstrument](#) type exposes the following members.

Properties

Name	Description
  DayInfo	Day bar information for an instrument (Inherited from Instrument)
  ExpirationDateUtc	The UTC expiry date of the contract
  FirstTradeDateUtc	The UTC date of first trade of the contract
  LastTradeDateUtc	The UTC date of last trade of the contract
  LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)

 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotSize	Amount of base asset for one lot
 	LotStep	Step of the lot changes (Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MaturityDateUtc	Contract month date in UTC
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	OptionType	Put/Call price type
 	StrikePrice	Strike prices which established when a contract is first written

	Symbol	Symbol of the base instrument (Inherited from Instrument)
	TickSize TickCost	Array of minimal value size of pair's quote used by script
	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
	Type	Market category of the instrument (Overrides InstrumentType)
	Underlayer	Underlayer Instrument

[Top](#)

◀ **See Also**

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentExpiration DateUtc Property

The UTC expiry date of the contract

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime  
ExpirationDateUtc { get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
using System;  
using TradeApi.Instruments;  
using System;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
ExpirationDateExample :  
StrategyBuilder  
{  
    public  
ExpirationDateExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"ExpirationDateExample";  
        #endregion  
    }  
    OrderRequest  
request;  
  
        public override  
void Init()  
    {  
        var instrument  
=  
HistoryDataSeries.HistoricalRe  
quest.Instrument;  
        if (instrument  
is OptionsInstrument)  
            request =  
new  
OrderRequest(OrderType.Market,  
instrument,  
AccountManager.Current,  
OrderSide.Buy, (instrument as  
OptionsInstrument).LotSize ??  
instrument.MinimalLot(ProductT  
ype.General));  
    }  
  
        public override  
void Update(TickStatus args)  
    {
```

```
        if (args ==  
    TickStatus.IsBar && request !=  
    null)  
    {  
        var option  
    = InstrumentsManager.Current  
    as OptionsInstrument;  
        var  
    ExpirationDate =  
    option.ExpirationDateUtc >  
    DateTime.UtcNow;  
  
        if (ExpirationDate)  
  
    OrdersManager.Send(request);  
    }  
    }  
}
```



See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentFirstTrade DateUtc Property

The UTC date of first trade of the contract

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime  
FirstTradeDateUtc { get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
FirstTradeDateExample :  
StrategyBuilder
```

[Copy](#)

```
        {
            public
FirstTradeDateExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"FirstTradeDateUtcExample";
    #endregion
}
OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
    if (instrument
is OptionsInstrument)
        request =
new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Buy, (instrument as
OptionsInstrument).LotSize ??
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request !=
```

```
    null)
    {
        var option
= InstrumentsManager.Current
as OptionsInstrument;

if(option.LastTradeDateUtc <
option.FirstTradeDateUtc)

OrdersManager.Send(request);
}
}
}
```



See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentLastTrade DateUtc Property

The UTC date of last trade of the contract

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime  
LastTradeDateUtc { get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
using System;  
using TradeApi.Instruments;  
using System;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
LastTradeDateExample :  
StrategyBuilder  
{  
    public  
LastTradeDateExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"LastTradeDateExample";  
        #endregion  
    }  
    OrderRequest  
request;  
  
        public override  
void Init()  
    {  
        var instrument  
=  
HistoryDataSeries.HistoricalRe  
quest.Instrument;  
        if (instrument  
is OptionsInstrument)  
            request =  
new  
OrderRequest(OrderType.Market,  
instrument,  
AccountManager.Current,  
OrderSide.Buy, (instrument as  
OptionsInstrument).LotSize ??  
instrument.MinimalLot(ProductT  
ype.General));  
    }  
  
        public override  
void Update(TickStatus args)  
    {
```

```
        if (args ==  
    TickStatus.IsBar && request !=  
    null)  
    {  
        var option  
    = InstrumentsManager.Current  
    as OptionsInstrument;  
  
        if(option.LastTradeDateUtc <  
    DateTime.UtcNow)  
  
            OrdersManager.Send(request);  
        }  
    }  
}
```



See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

OptionsInstrumentLot Size Property

Amount of base asset for one lot

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int? LotSize { get; } Copy
```

Property Value

[NullableInt32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
LotSizeExample :
StrategyBuilder
{
    public
LotSizeExample()
```

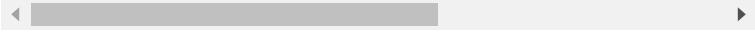
```
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"LotSizeExample";
        #endregion
    }
OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
    if(instrument
is OptionsInstrument)
        request =
new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy, (instrument as
OptionsInstrument).LotSize ??
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request != null)
{
```

```
        OrdersManager.Send(request);  
    }  
}  
}  
}
```



▲ See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentMaturity DateUtc Property

Contract month date in UTC

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime  
MaturityDateUtc { get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi.Instruments;  
using System;  
  
namespace TestEnv  
{  
    public class  
MaturityDateExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
MaturityDateExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"MaturityDateExample";
        #endregion
    }
    OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
        if(instrument
is OptionsInstrument)
            request =
new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy, (instrument as
OptionsInstrument).LotSize ??
instrument.MinimalLot(ProductT
ype.General));
    }

        public override
void Update(TickStatus args)
{
    if (args ==

```

```
    TickStatus.IsBar && request !=  
    null)  
    {  
        var option  
        = InstrumentsManager.Current  
        as OptionsInstrument;  
        var  
        isMatured =  
        option.MaturityDateUtc <  
        DateTime.UtcNow;  
  
        if (!isMatured)  
  
            OrdersManager.Send(request);  
        }  
    }  
}
```



See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentOptionType Property

Put/Call price type

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public OptionType OptionType {  
    get; }
```

[Copy](#)

Property Value

[OptionType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
        OptionTypeExample :  
            StrategyBuilder
```

[Copy](#)

```
        {
            public
OptionTypeExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"OptionTypeExample";
    #endregion
}
OrderRequest
request;

        public override
void Init()
{
    var instrument
=
HistoryDataSeries.HistoricalRe
quest.Instrument;
    if (instrument
is OptionsInstrument)
        request =
new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy, (instrument as
OptionsInstrument).LotSize ??
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar && request !=
```

```
    null)
    {
        var option
= InstrumentsManager.Current
as OptionsInstrument;

if(option.OptionType ==
OptionType.Put)

OrdersManager.Send(request);
}
}
}
```



See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentStrikePrice Property

Strike prices which established when a contract is first written

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double StrikePrice {  
    get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
OptionTypeExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
OptionTypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"StrikePriceExample";
    #endregion
}

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var option
= InstrumentsManager.Current
as OptionsInstrument;

Notification.Print($"option:
{option.Symbol}; stike price:
{option.StrikePrice}");
    }
}
}
```

See Also

[Reference](#)

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentTickSize TickCost Property

Array of minimal value size of pair's quote used by script

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public List<TicksizeTickcost>
    TickSizeTickCost { get; }
```

[Copy](#)

Property Value
[ListTicksizetickcost](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
        TickSizeTickCostExample :
            StrategyBuilder
```

[Copy](#)

```
        {
            public
TickSizeTickCostExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TickSizeTickCostExample";
    #endregion
}
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.Option)
    {
        var
tickSize =
(InstrumentsManager.Current as
OptionsInstrument).TickSizeTic
kCost;

tickSize.ForEach(element =>
Notification.Print($" tick
size: {element.TickSize}; tick
```

```
    cost: {element.TickCost}"));  
        }  
    }  
}
```

See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value

[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
```

[Copy](#)

```
        public
TypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TypeExample";
    #endregion
}
OrderRequest
request;
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot(ProductT
ype.General));
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.Option)
{
```

```
        OrdersManager.Send(request);  
    }  
}  
}  
}
```



▲ See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Options InstrumentUnderlayer Property

Underlayer Instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Underlayer {  
    get; }
```

[Copy](#)

Property Value

[Instrument](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
UnderlayerExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public
UnderlayerExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"UnderlayerExample";
    #endregion
}
Instrument
instrument;

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
TradeApi.Instruments.Instrumen
tType.Option)
    {
        var
underlayer =
(InstrumentsManager.Current as
OptionsInstrument).Underlayer?
.Symbol;

if(underlayer != string.Empty)

Notification.Print($" the
option instrument:
```

```
{ InstrumentsManager.Current.Sy  
mbol} has underlaying asset:  
{underlayer});  
}  
}  
}  
}
```

See Also

Reference

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

OptionsInstrument Methods

The [OptionsInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[OptionsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

OptionType Enumeration

Option type

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum OptionType
```

[Copy](#)

► Members

Member name	Value	Description
Put	0	Put
Call	1	Put

► See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

SpotInstrument Class

Spot instrument entity

▪ Inheritance Hierarchy

SystemObject [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsSpotInstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
SpotInstrument : Instrument
```

[Copy](#)

The [SpotInstrument](#) type exposes the following members.

▪ Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)
 LimitHigh	Gets the high price limit to the

		selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)
 	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)



Type

Market category
of the
instrument
(Overrides
[InstrumentType](#))

[Top](#)

Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot) (Inherited from Instrument)



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

See Also

Reference

[TradeApi.Instruments Namespace](#)

SpotInstrument Properties

The [SpotInstrument](#) type exposes the following members.

Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)

 MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 Symbol	Symbol of the base instrument (Inherited from Instrument)
 TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

See Also

[Reference](#)

[SpotInstrument Class](#)

[TradeApi.Instruments Namespace](#)

SpotInstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value
[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
    public
TypeExample()
```

[Copy](#)

```
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"TypeExample";
        #endregion
    }
    OrderRequest
request;
    Instrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot());
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.Spot)
    {

OrdersManager.Send(request);
    }
}
```

```
        }  
    }
```



See Also

Reference

[SpotInstrument Class](#)

[TradeApi.Instruments Namespace](#)

SpotInstrument Methods

The [SpotInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[SpotInstrument Class](#)

[TradeApi.Instruments Namespace](#)

SpreadBetInstrument Class

SpreadBet instrument entity

► Inheritance Hierarchy

SystemObject TradeApi.InstrumentsInstrument

TradeApi.InstrumentsSpreadBetInstrument

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
SpreadBetInstrument :  
Instrument
```

Copy

The SpreadBetInstrument type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument

		(Inherited from Instrument)
	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Symbol	Symbol of the base instrument (Inherited from Instrument)

 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

◀ Methods

	Name	Description
 	Equals	Allow to compare two instruments (Inherited from Instrument)
 	GetHashCode	(Inherited from Instrument)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)



MinimalLot

The smallest trade allowed (in lot)
(Inherited from [Instrument](#))



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

SpreadBetInstrument Properties

The [SpreadBetInstrument](#) type exposes the following members.

Properties

	Name	Description
 	DayInfo	Day bar information for an instrument (Inherited from Instrument)
 	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 	LotStep	Step of the lot changes (Inherited from Instrument)

 MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 Symbol	Symbol of the base instrument (Inherited from Instrument)
 TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

See Also

[Reference](#)

[SpreadBetInstrument Class](#)

[TradeApi.Instruments Namespace](#)

SpreadBet InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value

[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
```

[Copy](#)

```
    public
TypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TypeExample";
    #endregion
}
OrderRequest
request;
Instrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot());
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.SpreadBet)
{

OrdersManager.Send(request);
}
```

```
        }  
    }  
}
```



See Also

[Reference](#)

[SpreadBetInstrument Class](#)

[TradeApi.Instruments Namespace](#)

SpreadBetInstrument Methods

The [SpreadBetInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[SpreadBetInstrument Class](#)

[TradeApi.Instruments Namespace](#)

SyntheticInstrument Class

Synthetic instrument entity

► Inheritance Hierarchy

SystemObject TradeApi.InstrumentsInstrument
TradeApi.InstrumentsSyntheticInstrument

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class  
SyntheticInstrument : Instrument
```

Copy

The SyntheticInstrument type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)
 LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)

	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Symbol	Symbol of the base instrument (Inherited from Instrument)
	SyntheticInstrumentContents	List of sub instruments and coefficients
	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
	Type	Market category of the instrument

(Overrides
[InstrumentType](#))

[Top](#)

Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot) (Inherited from Instrument)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

TradeApi.Instruments Namespace

SyntheticInstrument Properties

The [SyntheticInstrument](#) type exposes the following members.

Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)
 LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 LotStep	Step of the lot changes (Inherited from Instrument)
 MarketDepth	MarketDepth info for an instrument

		(Inherited from Instrument)
 	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 	Symbol	Symbol of the base instrument (Inherited from Instrument)
 	SyntheticInstrumentContents	List of sub instruments and coefficients
 	TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 	Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

See Also

[Reference](#)

[SyntheticInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Synthetic InstrumentSynthetic InstrumentContents Property

List of sub instruments and coefficients

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public  
List<SyntheticInstrumentContent> SyntheticInstrumentContents  
{ get; }
```

[Copy](#)

Property Value

[ListSyntheticInstrumentContent](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi.Instruments;  
  
namespace TestEnv
```

[Copy](#)

```
    {
        public class
SyntheticInstrumentExample : 
StrategyBuilder
{
    public
SyntheticInstrumentExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SyntheticInstrumentExample";
        #endregion
    }
    Instrument
instrument;
}

        public override
void Init()
{
    instrument =
InstrumentsManager.Current;

if(instrument.Type ==
InstrumentType.Synthetic)
{
    var
synthetic = instrument as
SyntheticInstrument;

foreach(var content in
synthetic.SyntheticInstrumentC
ontents)
{
    Notification.Print($"Symbol:
{content.Instrument.Symbol}
Coefficient:
```

```
{content.Coefficient}");  
        }  
    }  
  
    public override  
    void Update(TickStatus args)  
    {  
    }  
}
```

See Also

[Reference](#)

[SyntheticInstrument Class](#)

[TradeApi.Instruments Namespace](#)

Synthetic InstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value

[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
SyntheticInstrumentExample :
StrategyBuilder
```

[Copy](#)

```
        {
            public
SyntheticInstrumentExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"SyntheticInstrumentExample";
    #endregion
}
Instrument
instrument;
public override
void Init()
{
    instrument =
InstrumentsManager.Current;

if(instrument.Type ==
InstrumentType.Synthetic)
{
    var
synthetic = instrument as
SyntheticInstrument;

foreach(var content in
synthetic.SyntheticInstrumentC
ontents)
{
    Notification.Print($"Symbol:
{content.Instrument.Symbol}
Coefficient:
{content.Coefficient}");
}
}
}
```

```
        public override  
void Update(TickStatus args)  
    {  
        }  
    }  
}
```

See Also

[Reference](#)

[SyntheticInstrument Class](#)

[TradeApi.Instruments Namespace](#)

SyntheticInstrument Methods

The [SyntheticInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[SyntheticInstrument Class](#)

[TradeApi.Instruments Namespace](#)

SyntheticInstrumentContent Class

SyntheticInstrumentContent entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.Instruments.SyntheticInstrumentContent](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
SyntheticInstrumentContent
```

[Copy](#)

The [SyntheticInstrumentContent](#) type exposes the following members.

► Properties

	Name	Description
 	Coefficient	Coefficient in synthetic instrument
 	Instrument	Sub instrument in synthetic instrument

[Top](#)

◀ Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

SyntheticInstrument Content Properties

The [SyntheticInstrumentContent](#) type exposes the following members.

Properties

	Name	Description
 	Coefficient	Coefficient in synthetic instrument
 	Instrument	Sub instrument in synthetic instrument

[Top](#)

See Also

[Reference](#)

[SyntheticInstrumentContent Class](#)

[TradeApi.Instruments Namespace](#)

SyntheticInstrument ContentCoefficient Property

Coefficient in synthetic instrument

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double Coefficient {  
    get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
        SyntheticInstrumentExample :  
            StrategyBuilder
```

[Copy](#)

```
        {
            public
SyntheticInstrumentExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"SyntheticInstrumentExample";
    #endregion
}
Instrument
instrument;
public override
void Init()
{
    instrument =
InstrumentsManager.Current;

if(instrument.Type ==
InstrumentType.Synthetic)
{
    var
synthetic = instrument as
SyntheticInstrument;

foreach(var content in
synthetic.SyntheticInstrumentC
ontents)
{
    Notification.Print($"Symbol:
{content.Instrument.Symbol}
Coefficient:
{content.Coefficient}");
}
}
}
```

```
        public override  
void Update(TickStatus args)  
    {  
    }  
}
```

See Also

Reference

[SyntheticInstrumentContent Class](#)

[TradeApi.Instruments Namespace](#)

SyntheticInstrument ContentInstrument Property

Sub instrument in synthetic instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Instrument {  
    get; }
```

[Copy](#)

Property Value

[Instrument](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
SyntheticInstrumentExample :  
StrategyBuilder
```

[Copy](#)

```
        {
            public
SyntheticInstrumentExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"SyntheticInstrumentExample";
    #endregion
}
Instrument
instrument;
public override
void Init()
{
    instrument =
InstrumentsManager.Current;

if(instrument.Type ==
InstrumentType.Synthetic)
{
    var
synthetic = instrument as
SyntheticInstrument;

foreach(var content in
synthetic.SyntheticInstrumentC
ontents)
{
    Notification.Print($"Symbol:
{content.Instrument.Symbol}
Coefficient:
{content.Coefficient}");
}
}
}
```

```
        public override  
void Update(TickStatus args)  
    {  
    }  
}
```

See Also

Reference

[SyntheticInstrumentContent Class](#)

[TradeApi.Instruments Namespace](#)

SyntheticInstrument Content Methods

The [SyntheticInstrumentContent](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[SyntheticInstrumentContent Class](#)

[TradeApi.Instruments Namespace](#)

TBillsIsnstrument Class

TBill instrument entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.InstrumentsInstrument](#)

[TradeApi.InstrumentsTBillsIsnstrument](#)

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class TBillsIsnstrument : Instrument
```

[Copy](#)

The [TBillsIsnstrument](#) type exposes the following members.

► Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)

	LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
	LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
	LotStep	Step of the lot changes (Inherited from Instrument)
	MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
	MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
	Symbol	Symbol of the base instrument (Inherited from Instrument)
	TradingStatus	Current trading status for the

instrument
(Inherited from
[Instrument](#))

 	Type	Market category of the instrument (Overrides InstrumentType)
---	----------------------	---

[Top](#)

◀ Methods

	Name	Description
 	Equals	Allow to compare two instruments (Inherited from Instrument)
 	GetHashCode	(Inherited from Instrument)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
 	MinimalLot	The smallest

trade allowed
(in lot)
(Inherited from
[Instrument](#))



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

TBillsInstrument Properties

The [TBillsInstrument](#) type exposes the following members.

Properties

Name	Description
 DayInfo	Day bar information for an instrument (Inherited from Instrument)
 LimitHigh	Gets the high price limit to the selected instrument (Inherited from Instrument)
 LimitLow	Gets the low price limit to the selected instrument (Inherited from Instrument)
 LotStep	Step of the lot changes (Inherited from Instrument)

 MarketDepth	MarketDepth info for an instrument (Inherited from Instrument)
 MinimalTickSize	Min value size of pair's quote used by script (Inherited from Instrument)
 Symbol	Symbol of the base instrument (Inherited from Instrument)
 TradingStatus	Current trading status for the instrument (Inherited from Instrument)
 Type	Market category of the instrument (Overrides InstrumentType)

[Top](#)

◀ See Also

[Reference](#)

[TBillsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

TBillsInstrumentType Property

Market category of the instrument

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override InstrumentType
Type { get; }
```

[Copy](#)

Property Value
[InstrumentType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.Instruments;
using TradeApi.Instruments;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
{
    public
TypeExample()
```

[Copy](#)

```
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
        "TypeExample";
        #endregion
    }
    OrderRequest
request;
    Instrument
instrument;

    public override
void Init()
{
    instrument =
InstrumentsManager.Current;
    request = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Sell,
instrument.MinimalLot());
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
instrument.Type ==
InstrumentType.TBill)
    {

OrdersManager.Send(request);
    }
}
```

```
        }  
    }
```



See Also

Reference

[TBillsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

TBillsInstrument Methods

The [TBillsInstrument](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Allow to compare two instruments (Inherited from Instrument)
	GetHashCode	(Inherited from Instrument)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MaximalLot	The biggest trade allowed (in lot) (Inherited from Instrument)
	MinimalLot	The smallest trade allowed (in lot)

(Inherited from
[Instrument](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TBillsInstrument Class](#)

[TradeApi.Instruments Namespace](#)

TickInterval Class

TickInterval entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.Instruments](#)
Tick
Interval

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
    TickInterval
```

[Copy](#)

The [TickInterval](#) type exposes the following members.

▪ Properties

	Name	Description
	LeftBorder	Left price boundary of the interval
	RightBorder	Right price boundary of the interval

[Top](#)

◀ Methods

	Name	Description
≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetType	Gets the Type of the current instance. (Inherited from Object)
≡	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

TickInterval Properties

The [TickInterval](#) type exposes the following members.

► Properties

	Name	Description
	LeftBorder	Left price boundary of the interval
	RightBorder	Right price boundary of the interval

[Top](#)

► See Also

[Reference](#)

[TickInterval Class](#)

[TradeApi.Instruments Namespace](#)

TickIntervalLeftBorder Property

Left price boundary of the interval

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double LeftBorder {  
    get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.Instruments;  
using System;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
LeftBorderExample :  
IndicatorBuilder  
    {  
        public
```

```
LeftBorderExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"LeftBorderExample";
    #endregion

base.SeparateWindow = false;
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;
    if (instrument
is FuturesInstrument)
{
    var list =
(instrument as
FuturesInstrument).TickSizeTic
kCost;

list.ForEach(inst => {

Notification.Comment($"Tick
interval
{list.IndexOf(inst)+1} [Tick
size :{inst.TickSize} / Tick
range :
```

```
{Math.Abs(inst.Interval.LeftBo  
rder -  
inst.Interval.RightBorder) } /  
Tick cost: {inst.TickCost}]");  
} );  
}  
}  
}
```

See Also

Reference

[TickInterval Class](#)

[TradeApi.Instruments Namespace](#)

TickIntervalRight Border Property

Right price boundary of the interval

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double RightBorder {  
    get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.Instruments;  
using System;  
  
namespace TestEnv  
{  
    public class  
RightBorderExample :  
IndicatorBuilder  
{
```

[Copy](#)

```
        public
RightBorderExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"RightBorderExample";
    #endregion

base.SeparateWindow = false;
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;
    if (instrument
is FuturesInstrument)
    {
        var list =
(instrument as
FuturesInstrument).TickSizeTic
kCost;

list.ForEach(inst => {

Notification.Comment($"Tick
interval
{list.IndexOf(inst)+1} [Tick
size :{inst.TickSize} / Tick
```

```
range :  
    {Math.Abs(inst.Interval.LeftBo  
rder -  
inst.Interval.RightBorder) } /  
    Tick cost: {inst.TickCost}]");  
        } );  
    }  
}
```

See Also

Reference

[TickInterval Class](#)

[TradeApi.Instruments Namespace](#)

TickInterval Methods

The [TickInterval](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TickInterval Class](#)

[TradeApi.Instruments Namespace](#)

Ticksize Class

Ticksize entity

► Inheritance Hierarchy

```
SystemObject TradeApi.InstrumentsTick  
size  
    TradeApi.InstrumentsTicksizeTickcost
```

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public class Ticksize
```

[Copy](#)

The `Ticksize` type exposes the following members.

► Properties

	Name	Description
	<code>Interval</code>	Instrument's price interval for specific tick cost
	<code>TickSize</code>	Min value size of pair's quote used by script

[Top](#)

◀ Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current

instance.
(Inherited
from [Object](#))



[MemberwiseClone](#)

Creates a shallow copy of the current [Object](#).
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

Ticksize Properties

The [Ticksize](#) type exposes the following members.

► Properties

	Name	Description
	Interval	Instrument's price interval for specific tick cost
	TickSize	Min value size of pair's quote used by script

[Top](#)

► See Also

[Reference](#)

[Ticksize Class](#)

[TradeApi.Instruments Namespace](#)

TicksizeInterval Property

Instrument's price interval for specific tick cost

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TickInterval Interval { Copy
    get; }
```

Property Value

[TickInterval](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.Instruments;
using System;

namespace TestEnv
{
    public class
IntervalExample :
IndicatorBuilder
```

```
        {
            public
IntervalExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"IntervalExample";
#endregion

base.SeparateWindow = false;
}

public override
void Init()
{

}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;
    if (instrument
is FuturesInstrument)
    {
        var list =
(instrument as
FuturesInstrument).TickSizeTic
kCost;

list.ForEach(inst => {

Notification.Comment($"Tick
interval
{list.IndexOf(inst)+1} [Tick
```

```
size :{inst.TickSize} / Tick  
range :  
{Math.Abs(inst.Interval.LeftBo  
rder -  
inst.Interval.RightBorder)} /  
Tick cost: {inst.TickCost]}");  
} );  
}  
}  
}
```

See Also

[Reference](#)

[Ticksize Class](#)

[TradeApi.Instruments Namespace](#)

TicksizeTickSize Property

Min value size of pair's quote used by script

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double TickSize { get;  
}
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
    TickSizeExample :  
    IndicatorBuilder  
    {  
        public  
        TickSizeExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"TickSizeExample";
        #endregion

base.SeparateWindow = false;
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;
    if (instrument
is FuturesInstrument)
{
    var list =
(instrument as
FuturesInstrument).TickSizeTic
kCost;

list.ForEach(inst =>
Notification.Comment($"Tick
cost is {inst.TickSize}"));
}
}
}
```

See Also

[Reference](#)

[Ticksize Class](#)

[TradeApi.Instruments Namespace](#)

Ticksize Methods

The [Ticksize](#) type exposes the following members.

▪ Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
 GetHashCode	Serves as the default

hash
function.
(Inherited
from [Object](#))



[GetType](#)

Gets the
[Type](#) of the
current
instance.
(Inherited
from [Object](#))



[MemberwiseClone](#)

Creates a
shallow copy
of the
current
[Object](#).
(Inherited
from [Object](#))



[ToString](#)

Returns a
string that
represents
the current
object.
(Inherited
from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[Ticksize Class](#)

[TradeApi.Instruments Namespace](#)

TicksizeTickcost Class

TicksizeTickcost entity

► Inheritance Hierarchy

SystemObject TradeApi.InstrumentsTicksize

TradeApi.InstrumentsTicksizeTickcost

Namespace: TradeApi.Instruments

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class  
    TicksizeTickcost : Ticksize
```

[Copy](#)

The TicksizeTickcost type exposes the following members.

► Properties

	Name	Description
	Interval	Instrument's price interval for specific tick cost (Inherited from Ticksize)
	TickCost	The cost of the

smallest price change in the quoted currency, per one lot Zero means TickCost is not specified and can be calculated as LotSize / TickSize



TickSize Min value size of pair's quote used by script
(Inherited from [Ticksize](#))

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Instruments Namespace](#)

Ticksizetickcost Properties

The [Ticksizetickcost](#) type exposes the following members.

Properties

	Name	Description
	Interval	Instrument's price interval for specific tick cost (Inherited from Ticksize)
	TickCost	The cost of the smallest price change in the quoted currency, per one lot Zero means TickCost is not specified and can be calculated as LotSize / TickSize
	TickSize	Min value size of pair's quote used by script (Inherited from Ticksize)

[Top](#)

See Also

Reference

[TicksizeTickcost Class](#)

[TradeApi.Instruments Namespace](#)

TicksizeTickcostTick Cost Property

The cost of the smallest price change in the quoted currency, per one lot Zero means TickCost is not specified and can be calculated as LotSize / TickSize

Namespace: [TradeApi.Instruments](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double TickCost { get;  
}
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.Instruments;  
  
namespace TestEnv  
{  
    public class  
    TickCostExample :  
    IndicatorBuilder
```

[Copy](#)

```
        {
            public
TickCostExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TickCostExample";
#endregion

base.SeparateWindow = false;
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    var instrument =
InstrumentsManager.Current;
    if (instrument
is FuturesInstrument)
    {
        var list =
(instrument as
FuturesInstrument).TickSizeTic
kCost;

list.ForEach(inst =>
Notification.Comment($"Tick
cost is {inst.TickCost}"));
    }
}
```

```
        }  
    }
```

See Also

Reference

[TicksizeTickcost Class](#)

[TradeApi.Instruments Namespace](#)

TicksizeTickcost Methods

The [TicksizeTickcost](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TicksizeTickcost Class](#)

[TradeApi.Instruments Namespace](#)

TradingStatus Enumeration

Trading status type

Namespace: [TradeApi.Instruments](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum TradingStatus
```

[Copy](#)

► Members

Member name	Value	Description
Close	0	Close
Open	1	Open
Halt	2	Halt

► See Also

Reference

[TradeApi.Instruments Namespace](#)

TradeApi.Quality Namespace

Classes

	Class	Description
	BaseQuote	BaseQuote entity
	BBO	Best bid offer quote
	Level2	Level2 quote
	Trade	Trade quote

Enumerations

	Enumeration	Description
	AggressorFlag	Aggressor flag type
	Level2Side	Level2 quote side type
	QuoteTypes	Source type of quotes

AggressorFlag Enumeration

Aggressor flag type

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum AggressorFlag
```

[Copy](#)

► Members

Member name	Value	Description
Ask	0	Ask
Bid	1	Bid
CrossTrade	2	CrossTrade
Auction	3	Auction
RLP	4	Retail Liquidity Provider
None	-1	None
NotSet	255	NotSet

See Also

Reference

[TradeApi.Quotes Namespace](#)

BaseQuote Class

BaseQuote entity

▪ Inheritance Hierarchy

```
SystemObject TradeApi.QuotesBaseQuote
  TradeApi.QuotesBBO
  TradeApi.QuotesLevel2
  TradeApi.QuotesTrade
```

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public abstract class
BaseQuote
```

[Copy](#)

The [BaseQuote](#) type exposes the following members.

▪ Properties

Name	Description
 Instrument	Represents the symbol of instrument for trade
 InternalInstrument	



TimeUtc

Time when trade occurred in UTC

[Top](#)

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
	GetHashCode	Serves as the default hash

function.
(Inherited
from [Object](#))

	GetType	Gets the Type of the current instance. (Inherited from Object)
---	-------------------------	--

	MemberwiseClone	Creates a shallow copy of the current Object . (Inherited from Object)
---	---------------------------------	--

	ToString	Returns a string that represents the current object. (Inherited from Object)
---	--------------------------	--

[Top](#)

See Also

[Reference](#)

[TradeApi.Quotes Namespace](#)

BaseQuote Properties

The [BaseQuote](#) type exposes the following members.

► Properties

Name	Description
 Instrument	Represents the symbol of instrument for trade
 InternalInstrument	
 TimeUtc	Time when trade occurred in UTC

[Top](#)

► See Also

[Reference](#)

[BaseQuote Class](#)

[TradeApi.Quotes Namespace](#)

BaseQuoteInstrument Property

Represents the symbol of instrument for trade

Namespace: [TradeApi.Quality](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Instrument { Copy
    get; }
```

Property Value

[Instrument](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
InstrumentExample :
StrategyBuilder
{
    public
```

```
InstrumentExample()
    : base()
{
    #region
    Initialization

    Credentials.ProjectName =
    "InstrumentExample";
    #endregion
}

        public override
void Init()
{
    InstrumentsManager.OnBBO +=
    (bbo) => {

        Notification.Comment($"Instrument's symbol is
{bbo.Instrument.Symbol}");
    };
}

        public override
void Update(TickStatus args)
{
    }
}
```

See Also

[Reference](#)

[BaseQuote Class](#)

[TradeApi.Quotes Namespace](#)

BaseQuoteInternal Instrument Property

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
protected [T:5zA=.8Ng=]  
InternalInstrument { get; }
```

[Copy](#)

Property Value

[T:5zA=.8Ng=]

► See Also

[Reference](#)

[BaseQuote Class](#)

[TradeApi.Quotes Namespace](#)

BaseQuoteTimeUtc Property

Time when trade occurred in UTC

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime TimeUtc { get; } Copy
```

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Trading;
using TradeApi.History;

namespace TestEnv
{
    public class
TimeUtcExample :
StrategyBuilder
{
    public
TimeUtcExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "TimeUtcExample";
        #endregion
    }

        public override
void Init()
{
    InstrumentsManager.OnBBO +=
(bbo) => {

    Notification.Comment($"Instrument's quote time in UTC is
{bbo.TimeUtc.ToString("O")}");
};

    InstrumentsManager.OnLevel2 +=
(level2) => {

    Notification.Comment($"Instrument's quote time in UTC is
{level2.TimeUtc.ToString("O")}");
};

    InstrumentsManager.OnTrades +=
(trade) => {

    Notification.Comment($"Instrument's quote time in UTC is
{trade.TimeUtc.ToString("O")}");
};
}
```

```
        g() }" );
    }
}

public override
void Update(TickStatus args)
{
    }

}
}
```

See Also

[Reference](#)

[BaseQuote Class](#)

[TradeApi.Quotes Namespace](#)

BaseQuote Methods

The [BaseQuote](#) type exposes the following members.

▪ Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
 GetHashCode	Serves as the default

hash
function.
(Inherited
from [Object](#))



[GetType](#)

Gets the
[Type](#) of the
current
instance.
(Inherited
from [Object](#))



[MemberwiseClone](#)

Creates a
shallow copy
of the
current
[Object](#).
(Inherited
from [Object](#))



[ToString](#)

Returns a
string that
represents
the current
object.
(Inherited
from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[BaseQuote Class](#)

[TradeApi.Quotes Namespace](#)

BBO Class

Best bid offer quote

► Inheritance Hierarchy

SystemObject TradeApi.QuotesBaseQuote
TradeApi.QuotesBBO

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class BBO :  
    BaseQuote
```

[Copy](#)

The [BBO](#) type exposes the following members.

► Properties

	Name	Description
	Ask	The best ask price
	AskSize	The volume of the inside ask
	Bid	The best bid price
	BidSize	The volume of the inside bid

	Instrument	Represents the symbol of instrument for trade (Inherited from BaseQuote)
	TimeUtc	Time when trade occurred in UTC (Inherited from BaseQuote)

[Top](#)

◀ Methods

Name	Description
	Equals Allow to compare two BBO (Overrides Object.Equals(Object))
	GetHashCode (Overrides Object.GetHashCode)
	GetType Gets the Type of the current instance. (Inherited from Object)
	ToString Returns a string that represents the current object. (Inherited from Object)

[Top](#)

Operators

	Name	Description
 S	Equality(BBO, BBO)	Allow to compare two BBO
 S	Inequality(BBO, BBO)	Allow to compare two BBO

[Top](#)

See Also

[Reference](#)

[TradeApi.Quotes Namespace](#)

BBO Properties

The [BBO](#) type exposes the following members.

► Properties

	Name	Description
 	Ask	The best ask price
 	AskSize	The volume of the inside ask
 	Bid	The best bid price
 	BidSize	The volume of the inside bid
 	Instrument	Represents the symbol of instrument for trade (Inherited from BaseQuote)
 	TimeUtc	Time when trade occurred in UTC (Inherited from BaseQuote)

[Top](#)

► See Also

Reference

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBOAsk Property

The best ask price

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Ask { get; }
```

[Copy](#)

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class AskExample
        : StrategyBuilder
    {
        public AskExample()
            : base()
        {
            #region
            Initialization
            Credentials.ProjectName =
        }
    }
}
```

[Copy](#)

```
"AskExample";
    #endregion
}

        public override
void Init()
{
    InstrumentsManager.OnBBO +=
(bbo) => {

    Notification.Comment($"Instrument's {bbo.Instrument.Symbol}
ask is {bbo.Ask}");
};

}

        public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBOAskSize Property

The volume of the inside ask

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double AskSize { get; } Copy
```

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
AskSizeExample :
StrategyBuilder
{
    public
AskSizeExample()
        : base()
    {
        #region
Initialization
    }
}
```

```
Credentials.ProjectName =
"AskSizeExample";
    #endregion
}

        public override
void Init()
{
    InstrumentsManager.OnBBO +=
(bbo) => {

    Notification.Comment($"Instrument's {bbo.Instrument.Symbol}
ask volume is {bbo.AskSize}");
};

}

        public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBOBid Property

The best bid price

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Bid { get; }
```

[Copy](#)

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class BidExample
        : StrategyBuilder
    {
        public BidExample()
            : base()
        {
            #region
            Initialization
            Credentials.ProjectName =
        }
    }
}
```

[Copy](#)

```
"BidExample";
    #endregion
}

        public override
void Init()
{
    InstrumentsManager.OnBBO +=
(bbo) => {

    Notification.Comment($"Instrument's {bbo.Instrument.Symbol}
bid is {bbo.Bid}");
}
}

        public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBOBidSize Property

The volume of the inside bid

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double BidSize { get; } Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
BidSizeExample :
StrategyBuilder
{
    public
BidSizeExample()
        : base()
    {
        #region
Initialization
    }
}
```

```
Credentials.ProjectName =
"BidSizeExample";
#endregion
}

public override
void Init()
{
    InstrumentsManager.OnBBO +=
(bbo) => {

    Notification.Comment($"Instrument's {bbo.Instrument.Symbol} 
bid volume is {bbo.BidSize}");
};

}

public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBO Methods

The [BBO](#) type exposes the following members.

► Methods

	Name	Description
	Equals	Allow to compare two BBO (Overrides Object.Equals(Object))
	GetHashCode	(Overrides Object.GetHashCode)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

► See Also

[Reference](#)

[BBO Class](#)

TradeApi.Quotes Namespace

BBOEquals Method

Allow to compare two BBO

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override bool Equals(Copy
                           Object obj
                           )
```

Parameters

obj [Object](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBOGetHashCode Method

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public override int  
GetHashCode()
```

[Copy](#)

Return Value

[Int32](#)

◀ See Also

[Reference](#)

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBO Operators

The [BBO](#) type exposes the following members.

▪ Operators

	Name	Description
 	Equality(BBO, BBO)	Allow to compare two BBO
 	Inequality(BBO, BBO)	Allow to compare two BBO

[Top](#)

▪ See Also

[Reference](#)

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBOEquality Operator

Allow to compare two BBO

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public static bool operator == Copy
(
    BBO bb01,
    BBO bbo2
)
```

Parameters

bb01 [BBO](#)

bbo2 [BBO](#)

Return Value

[Boolean](#)

◀ See Also

[Reference](#)

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

BBOInequality Operator

Allow to compare two BBO

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator !=  
(  
    BBO bbo1,  
    BBO bbo2  
)
```

[Copy](#)

Parameters

bbo1 [BBO](#)

bbo2 [BBO](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[BBO Class](#)

[TradeApi.Quotes Namespace](#)

Level2 Class

Level2 quote

▪ Inheritance Hierarchy

SystemObject TradeApi.QuotesBaseQuote
TradeApi.QuotesLevel2

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class Level2 :  
    BaseQuote
```

[Copy](#)

The `Level2` type exposes the following members.

▪ Properties

	Name	Description
 	Instrument	Represents the symbol of instrument for trade (Inherited from BaseQuote)

	MPID	ECN or exchange, where the orders are set
	NumberOfOrders	Number of orders
	Price	Price of Level2 quote
	Side	Side of Level2 quote
	Size	Size of Level2
	Source	Source of quote
	TimeUtc	Time when trade occurred in UTC (Inherited from BaseQuote)

[Top](#)

Methods

Name	Description
	Equals Allow to compare two Level2 quotes (Overrides Object.Equals(Object))



GetHashCode

(Overrides
[Object.GetHashCode](#))



GetType

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

Operators

	Name	Description
	Equality(Level2, Level2)	Allow to compare two Level2 quotes
	Inequality(Level2, Level2)	Allow to compare two Level2 quotes

[Top](#)

See Also

Reference

[TradeApi.Quotes Namespace](#)

Level2 Properties

The [Level2](#) type exposes the following members.

Properties

	Name	Description
 	Instrument	Represents the symbol of instrument for trade (Inherited from BaseQuote)
 	MPID	ECN or exchange, where the orders are set
 	NumberOfOrders	Number of orders
 	Price	Price of Level2 quote
 	Side	Side of Level2 quote
 	Size	Size of Level2

	Source	Source of quote
	TimeUtc	Time when trade occurred in UTC (Inherited from BaseQuote)

[Top](#)

◀ See Also

[Reference](#)

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2MPID Property

ECN or exchange, where the orders are set

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string MPID { get; }
```

[Copy](#)

Property Value
[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
MPIDExample : StrategyBuilder
    {
        public
MPIDExample()
        : base()
        {
            #region
Initialization
```

[Copy](#)

```
Credentials.ProjectName =
    "MPIDExample";
        #endregion
    }

        public override
void Init()
{
    InstrumentsManager.OnLevel2 +=
(level2) => {

    Notification.Comment($"Exchange's name is {level2.MPID}");
        };
}

        public override
void Update(TickStatus args)
{
    }
}
```

See Also

Reference

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2NumberOfOrders Property

Number of orders

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int NumberOfOrders {  
    get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
MPIDExample : StrategyBuilder  
    {  
        public  
MPIDExample()  
            : base()  
        {
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"NumberOfOrdersExample";
        #endregion
    }

        public override
void Init()
{
    InstrumentsManager.OnLevel2 +=
(level2) => {

    if (level2.NumberOfOrders > 0)

Notification.Comment($"Instrument's
{level2.Instrument.Symbol} is
ready to trade");
    };
}

        public override
void Update(TickStatus args)
{
    }
}
```

See Also

Reference

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2Price Property

Price of Level2 quote

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Price { get; } Copy
```

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
PriceExample : StrategyBuilder
    {
        public
PriceExample()
        : base()
        {
            #region
Initialization
```

```
Credentials.ProjectName =  
    "PriceExample";  
        #endregion  
    }  
  
    public override  
void Init()  
{  
  
    InstrumentsManager.OnLevel2 +=  
    (level2) => {  
  
        Notification.Comment($"Level2'  
s price is {level2.Price}");  
    };  
}  
  
    public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

Reference

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2Side Property

Side of Level2 quote

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Level2Side Side { get; }
```

Copy

Property Value

Level2Side

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
SideExample : StrategyBuilder
    {
        public
SideExample()
            : base()
        {
            #region
Initialization
        }
    }
}
```

Copy

```
Credentials.ProjectName =
"SideExample";
#endregion
}

public override
void Init()
{
    InstrumentsManager.OnLevel2 +=
(level2) => {

    if(level2.Side >
TradeApi.Quotes.Level2Side.Ask
)

Notification.Comment($"Level2'
s price is {level2.Price}");
};

}

public override
void Update(TickStatus args)
{
}

}
}
```

See Also

[Reference](#)

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2Size Property

Size of Level2

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Size { get; }
```

[Copy](#)

Property Value

Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
SizeExample : StrategyBuilder
    {
        public
SizeExample()
        : base()
        {
            #region
Initialization
        }
    }
}
```

[Copy](#)

```
Credentials.ProjectName =  
    "SizeExample";  
        #endregion  
    }  
  
    public override  
void Init()  
{  
  
    InstrumentsManager.OnLevel2 +=  
    (level2) => {  
  
        Notification.Comment($"Level2'  
        s size is {level2.Size}");  
    };  
}  
  
    public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

Reference

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2Source Property

Source of quote

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Source { get; } Copy
```

Property Value

String

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
SourceExample :
StrategyBuilder
{
    public
SourceExample()
        : base()
    {
        #region
Initialization
    }
}
```

```
Credentials.ProjectName =
"SourceExample";
#endregion
}

public override
void Init()
{
    InstrumentsManager.OnLevel2 +=
(level2) => {

    Notification.Comment($"Level2
qoute's source is
{level2.Source}");
};

}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2 Methods

The [Level2](#) type exposes the following members.

► Methods

Name	Description
Equals	Allow to compare two Level2 quotes (Overrides Object.Equals(Object))
GetHashCode	(Overrides Object.GetHashCode)
GetType	Gets the Type of the current instance. (Inherited from Object)
ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

► See Also

[Reference](#)

[Level2 Class](#)

TradeApi.Quotes Namespace

Level2Equals Method

Allow to compare two Level2 quotes

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override bool Equals(Copy
                           Object obj
                           )
```

Parameters

obj [Object](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2GetHashCode Method

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override int  
GetHashCode()
```

[Copy](#)

Return Value

[Int32](#)

► See Also

[Reference](#)

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2 Operators

The [Level2](#) type exposes the following members.

▪ Operators

	Name	Description
 S	Equality(Level2, Level2)	Allow to compare two Level2 quotes
 S	Inequality(Level2, Level2)	Allow to compare two Level2 quotes

[Top](#)

▪ See Also

[Reference](#)

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2Equality Operator

Allow to compare two Level2 quotes

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator ==  
(  
    Level2 quote1,  
    Level2 quote2  
)
```

[Copy](#)

Parameters

quote1 [Level2](#)
quote2 [Level2](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2Inequality Operator

Allow to compare two Level2 quotes

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator !=  
(  
    Level2 quote1,  
    Level2 quote2  
)
```

[Copy](#)

Parameters

quote1 [Level2](#)
quote2 [Level2](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Level2 Class](#)

[TradeApi.Quotes Namespace](#)

Level2Side Enumeration

Level2 quote side type

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum Level2Side
```

[Copy](#)

► Members

Member name	Value	Description
Bid	0	Bid
Ask	1	Ask

► See Also

[Reference](#)

[TradeApi.Quotes Namespace](#)

QuoteTypes Enumeration

Source type of quotes

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
[FlagsAttribute]  
public enum QuoteTypes
```

[Copy](#)

► Members

Member name	Value	Description
Quote	1	Quote
Level2	2	Level2
Trade	4	Trade

► See Also

[Reference](#)

[TradeApi.Quotes Namespace](#)

Trade Class

Trade quote

▪ Inheritance Hierarchy

SystemObject TradeApi.QuotesBaseQuote
TradeApi.QuotesTrade

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class Trade :  
    BaseQuote
```

[Copy](#)

The `Trade` type exposes the following members.

▪ Properties

	Name	Description
	Aggressor	Provides aggressor's value
	Buyer	Buyer's name
	Instrument	Represents the symbol of instrument for trade

(Inherited from
[BaseQuote](#))

 	Price	Returns the price at which trade occurred
 	Seller	Seller's name
 	Size	Value of the trade in lots
 	TimeUtc	Time when trade occurred in UTC (Inherited from BaseQuote)

[Top](#)

◀ Methods

Name	Description
 Equals	Allow to compare two Trade quotes (Overrides Object.Equals(Object))
 GetHashCode	(Overrides Object.GetHashCode)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

[Top](#)

Operators

	Name	Description
 S	Equality(Trade, Trade)	Allow to compare two Trade quotes
 S	Inequality(Trade, Trade)	Allow to compare two Trade quotes

[Top](#)

See Also

[Reference](#)

[TradeApi.Quotes Namespace](#)

Trade Properties

The [Trade](#) type exposes the following members.

Properties

	Name	Description
 	Aggressor	Provides aggressor's value
 	Buyer	Buyer's name
 	Instrument	Represents the symbol of instrument for trade (Inherited from BaseQuote)
 	Price	Returns the price at which trade occurred
 	Seller	Seller's name
 	Size	Value of the trade in lots
 	TimeUtc	Time when trade occurred in UTC (Inherited from BaseQuote)

[Top](#)

See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradeAggressor Property

Provides aggressor's value

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public AggressorFlag Aggressor
{ get; }
```

[Copy](#)

Property Value
[AggressorFlag](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
AggressorExample :
StrategyBuilder
{
    public
AggressorExample()
    : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "AggressorExample";  
    #endregion  
}  
  
    public override  
void Init()  
{  
  
    InstrumentsManager.OnTrades +=  
    (trade) => {  
  
        if(trade.Aggressive >  
        TradeApi.Quotes.AggressiveFlag.  
        Ask)  
  
            Notification.Comment(${Instrument  
            {trade.Instrument.Symbol}  
            is bought});  
    };  
}  
  
    public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradeBuyer Property

Buyer's name

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Buyer { get; } Copy
```

Property Value

String

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System;

namespace TestEnv
{
    public class
BuyerExample : StrategyBuilder
    {
        public
BuyerExample()
            : base()
        {
            #region
Initialization
        }
    }
}
```

```
Credentials.ProjectName =
"BuyerExample";
    #endregion
}

public override
void Init()
{
    InstrumentsManager.OnTrades +=
(trade) => {

    if (!String.IsNullOrEmpty(trade
.Buyer))

        Notification.Comment($"Last
trade buyer is
{trade.Buyer}");
    }
}

public override
void Update(TickStatus args)
{
    }
}
```

See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradePrice Property

Returns the price at which trade occurred

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Price { get; } Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
PriceExample : StrategyBuilder
    {
        public
PriceExample()
        : base()
        {
            #region
Initialization
        }
    }
}
```

```
Credentials.ProjectName =  
    "PriceExample";  
        #endregion  
    }  
  
    public override  
void Init()  
{  
  
    InstrumentsManager.OnTrades +=  
    (trade) => {  
  
        Notification.Comment($"Last  
        trade price is  
        {trade.Price}");  
    };  
}  
  
    public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradeSeller Property

Seller's name

Namespace: TradeApi.Quotes

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Seller { get; } Copy
```

Property Value

String

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System;

namespace TestEnv
{
    public class
SellerExample :
StrategyBuilder
{
    public
SellerExample()
    : base()
{
    #region
```

```
Initialization

Credentials.ProjectName =
"SellerExample";
#endregion
}

public override
void Init()
{

InstrumentsManager.OnTrades +=
(trade) => {

if (!String.IsNullOrEmpty(trade
.Seller))

Notification.Comment($"Last
trade seller is
{trade.Seller}");
};

}

public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradeSize Property

Value of the trade in lots

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Size { get; }
```

[Copy](#)

Property Value
Double

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
SizeExample : StrategyBuilder
    {
        public
SizeExample()
        : base()
        {
            #region
Initialization
```

[Copy](#)

```
Credentials.ProjectName =  
    "SizeExample";  
        #endregion  
    }  
  
    public override  
void Init()  
{  
  
    InstrumentsManager.OnTrades +=  
(trade) => {  
  
    Notification.Comment($"Last  
    trade lot is {trade.Size}");  
    };  
}  
  
    public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```

See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

Trade Methods

The [Trade](#) type exposes the following members.

► Methods

Name	Description
Equals	Allow to compare two Trade quotes (Overrides Object.Equals(Object))
GetHashCode	(Overrides Object.GetHashCode)
GetType	Gets the Type of the current instance. (Inherited from Object)
ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

► See Also

[Reference](#)

[Trade Class](#)

TradeApi.Quotes Namespace

TradeEquals Method

Allow to compare two Trade quotes

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override bool Equals(Copy
                           Object obj
                           )
```

Parameters

obj [Object](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradeGetHashCode Method

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override int  
GetHashCode()
```

[Copy](#)

Return Value

[Int32](#)

► See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

Trade Operators

The [Trade](#) type exposes the following members.

▪ Operators

	Name	Description
 	Equality(Trade, Trade)	Allow to compare two Trade quotes
 	Inequality(Trade, Trade)	Allow to compare two Trade quotes

[Top](#)

▪ See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradeEquality Operator

Allow to compare two Trade quotes

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator ==  
(  
    Trade quote1,  
    Trade quote2  
)
```

[Copy](#)

Parameters

quote1 [Trade](#)
quote2 [Trade](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradeInequality Operator

Allow to compare two Trade quotes

Namespace: [TradeApi.Quotes](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator !=  
(  
    Trade quote1,  
    Trade quote2  
)
```

[Copy](#)

Parameters

quote1 [Trade](#)
quote2 [Trade](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Trade Class](#)

[TradeApi.Quotes Namespace](#)

TradeApi.ToolBelt Namespace

► Classes

	Class	Description
	BaseScript	Base class dedicated to scripts to be developed for the platform
	Connection	Connection entity
	GlobalVariable	Global variable entry
	GlobalVariableManager	GlobalVariableManager entity
	Notification	Notification batch entity
	ScreenCapture	ScreenCapture entity
	Statistic	Statistic entity
	TimeConvertor	TimeConvertor entity
	Tools	Tools entity

► Enumerations

	Enumeration	Description
	ConnectionLossBehavior	ConnectionLossBehavior types
	ConnectionState	ConnectionState type

	ConnectionStatus	ConnectionStatus types
	VariableLifetime	VariableLifetime types

BaseScript Class

Base class dedicated to scripts to be developed for the platform

► Inheritance Hierarchy

```
SystemObject  ScriptBase
CSharpScript
CSharpIndicator
    TradeApi.ToolBeltBaseScript
    TradeApi.IndicatorsIndicatorBuilder
    TradeApi.TradingStrategyBuilder
```

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public abstract class BaseScript : CSharpIndicator
```

[Copy](#)

The `BaseScript` type exposes the following members.

► Properties

	Name	Description
	AccountManager	A manager reference to retrieve accounts
	Connection	Connection entity
	Credentials	Script credentials (Inherited from CSharpScript)
	Disposed	(Inherited from CSharpScript)
	HistoricalDataManager	A historicaldata manager reference to wield custom control over extra required historical element
	HistoryDataSeries	Represent access to instrument's historical data based on current chart
	Id	(Inherited from CSharpScript)
	IndicatorsManager	An indicator manager reference to obtain usage of buildin and custom indicators
	InstrumentsManager	A manager reference to query and manage vendor's instruments

	Notification	An indicator's notification hub
	ProjectType	(Inherited from CSharpScript)
	SeparateWindow	(Inherited from CSharpIndicator)
	SingleThreadEvents	(Inherited from CSharpScript)
	Statistic	Statistic entity
	Tools	Tool assistance in platform

[Top](#)

Methods

Name	Description
Complete	This function is called before running script (Overrides CSharpIndicator.Complete .)
Dispose	(Inherited from CSharpScript)
Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
GetArray(ArrayList)	(Inherited from ScriptBase)
GetArray(ArrayList, Boolean)	(Inherited from ScriptBase)
GetArray(Type, Int32)	(Inherited from ScriptBase)
GetHashCode	Serves as the default hash function. (Inherited from Object)
GetType	Gets the Type of the current instance. (Inherited from Object)
Init	This function is called once before work starts (Overrides CSharpIndicator.Init .)
LoadAndInvoke	(Inherited from CSharpScript)

 MemberwiseClone	Creates a shallow copy of the Object . (Inherited from Object)
 OnDebug	(Inherited from CSharpScript)
 OnPaintChart	(Inherited from CSharpScript)
 PushMethodDump	(Inherited from CSharpScript)
 QueueUserWorkItem(WaitCallback)	(Inherited from CSharpScript)
 QueueUserWorkItem(WaitCallback, Object)	(Inherited from CSharpScript)
 QueueUserWorkItem(WaitCallback, Object, ApartmentState)	(Inherited from CSharpScript)
 RegisterLocalVariable(String, FuncObject, String)	(Inherited from CSharpScript)
 RegisterLocalVariable(String, FuncObject, ActionObject, String)	(Inherited from CSharpScript)
 ToString	Returns a string that represents current object. (Inherited from Object)
 Update	This function is called amid new bar or history processing. argument with a tick status to quote state (args.TickStatus, IsHistory/IsB (Overrides CSharpIndicator.Update(TickEventArgs))

[Top](#)

Fields

Name	Description
 currentInternalInstrument	
 ScriptCache	(Inherited from CSharpScript)

[Top](#)

See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

BaseScript Properties

The [BaseScript](#) type exposes the following members.

Properties

Name	Description
 AccountManager	A manager reference to retrieve accounts
 Connection	Connection entity
 Credentials	Script credentials (Inherited from CSharpScript)
 Disposed	(Inherited from CSharpScript)
 HistoricalDataManager	A historicaldata manager reference to wield custom control over extra required historical element
 HistoryDataSeries	Represent access to instrument's historical data based on current chart
 Id	

(Inherited from
CSharpScript)



[IndicatorsManager](#)

An indicator manager reference to obtain usage of buildin and custom indicators



[InstrumentsManager](#)

A manager reference to query and manage vendor's instruments



[Notification](#)

An indicator's notification hub



ProjectType

(Inherited from
CSharpScript)



SeparateWindow

(Inherited from
CSharpIndicator)



SingleThreadEvents

(Inherited from
CSharpScript)



[Statistic](#)

Statistic entity



[Tools](#)

Tool assistance in platform

[Top](#)

◀ See Also

Reference

BaseScript Class
TradeApi.ToolBelt Namespace

BaseScriptAccount Manager Property

A manager reference to retrieve accounts

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public AccountManager  
AccountManager { get; }
```

[Copy](#)

Property Value
[AccountManager](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
AccountManagerExample :  
IndicatorBuilder  
    {  
        public  
AccountManagerExample()
```

[Copy](#)

```
        : base()
    {
        #region Initialization

        Credentials.ProjectName =
"AccountManagerExample";
        #endregion

        Lines.Set("AccountManagerExample");
    }

    base.SeparateWindow = false;
}

public override void Init()
{
}

public override void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var account =
AccountManager.GetAccounts(x
=> x.Status ==
AccountStatus.Active);

        account.ForEach(x =>
Notification.Comment(x.Name));
    }
}
}
```

See Also

Reference

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptConnection Property

Connection entity

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public Connection Connection {  
    get; }
```

[Copy](#)

Property Value
[Connection](#)

► See Also

Reference

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptHistorical DataManager Property

A historicaldata manager reference to wield custom control over extra required historical element

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public HistoricalDataManager  
HistoricalDataManager { get; }
```

[Copy](#)

Property Value
[HistoricalDataManager](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using System;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
HistoricalDataManagerExample :  
IndicatorBuilder
```

[Copy](#)

```
        {
            public
HistoricalDataManagerExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"HistoricalDataManagerExample"
;
#endregion

Lines.Set("HistoricalDataManagerExample");
}

base.SeparateWindow = false;
}

[InputParameter(InputType.Numeric, "Period of Moving
Average", 0)]
[SimpleNumeric(1D,
99999D)]
public int MAPeriod
= 1;

        HistoricalData
LoadedHistory;

        public override
void Init()
{
    var
historicalReq = new
TimeHistoricalRequest(Instrume
ntsManager.Current,
```

```
    HistoryDataSeries.HistoricalRe
quest.DataType, Period.Day,
MAPeriod);

HistoricalDataManager.OnLoaded
+=
this.HistoricalDataManager_OnL
oaded;

HistoricalDataManager.Get(hist
oricalReq, new
Interval(DateTime.UtcNow.AddDays(-1), DateTime.UtcNow));
}

private void
HistoricalDataManager_OnLoaded
(HistoricalData
loadedHistoricalData)
{
    LoadedHistory =
loadedHistoricalData;
    if
(LoadedHistory != null)
{
    for (int i
= LoadedHistory.Count - 1; i
>= 0; i--)
{
    var
open =
LoadedHistory.GetValue(PriceTy
pe.Open, i);
    var
high =
LoadedHistory.GetValue(PriceTy
pe.High, i);
    var low
=
LoadedHistory.GetValue(PriceTy
```

```
    pe.Low, i);
                                var
    close =
    LoadedHistory.GetValue(PriceTy
    pe.Close, i);

    Notification.Print(string.Format
        at("Open: {0}, High: {1}, Low:
        {2}, Close: {3}", open, high,
        low, close));
    }
}
}

public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptHistoryData Series Property

Represent access to instrument's historical data based on current chart

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

▪ Syntax

C#

```
public HistoricalData  
HistoryDataSeries { get; }
```

[Copy](#)

Property Value

[HistoricalData](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
HistoryDataSeriesExample :  
IndicatorBuilder  
    {  
        public
```

[Copy](#)

```
HistoryDataSeriesExample()
    : base()
{
    #region
    Initialization

    Credentials.ProjectName =
    "HistoryDataSeriesExample";
    #endregion

    Lines.Set("HistoryDataSeriesEx
    ample");

    base.SeparateWindow = false;
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if
(HistoryDataSeries.Count > 0)

    HistoryDataSeries.GetValue(PriceType.Close);
}
}
```

See Also

Reference

BaseScript Class
TradeApi.ToolBelt Namespace

BaseScriptIndicators Manager Property

An indicator manager reference to obtain usage of buildin and custom indicators

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public IndicatorsManager  
IndicatorsManager { get; }
```

[Copy](#)

Property Value
[IndicatorsManager](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
IndicatorsManagerExample :  
IndicatorBuilder  
    {  
        public  
IndicatorsManagerExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "IndicatorsManagerExample";
        #endregion

        Lines.Set("IndicatorsManagerEx
ample");

        base.SeparateWindow = false;
    }

    [InputParameter(InputType.Nume
ric, "Period of Moving
Average", 0)]
    [SimpleNumeric(1D,
99999D)]
    public int MAPeriod
= 13;

    public override
void Init()
{
    var buildIn =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 2,
MAMode.EMA);
    var custom =
IndicatorsManager.CreateInstance(this, new object[] {
        "MAPeriod",
5}, HistoryDataSeries);
}
```

```
        public override
void Update(TickStatus args)
{
    }

}
```

See Also

[Reference](#)

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

Base ScriptInstruments Manager Property

A manager reference to query and manage vendor's instruments

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public InstrumentsManager  
InstrumentsManager { get; }
```

[Copy](#)

Property Value
[InstrumentsManager](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
InstrumentsManagerExample :  
IndicatorBuilder  
{
```

[Copy](#)

```
        public
InstrumentsManagerExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"InstrumentsManagerExample";
#endregion

Lines.Set("InstrumentsManagerE
xample");

base.SeparateWindow = false;
}

        public override
void Init()
{
    InstrumentsManager.OnBBO +=
(BBO) => {

Notification.Print(BBO.Bid.ToString());
};

        public override
void Update(TickStatus args)
{
}

}
```

See Also

[Reference](#)

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptNotification Property

An indicator's notification hub

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public Notification  
Notification { get; }
```

[Copy](#)

Property Value
[Notification](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
NotificationExample :  
IndicatorBuilder  
    {  
        public  
NotificationExample()
```

[Copy](#)

```
: base()
{
    #region
Initialization

Credentials.ProjectName =
"NotificationExample";
    #endregion

Lines.Set("NotificationExample
");

base.SeparateWindow = false;
}

public override
void Init()
{
    const string
SCRIPT_START = "script
starts";

    switch
(DateTime.UtcNow.DayOfWeek) {
        case
DayOfWeek.Monday:

Notification.Alert(SCRIPT_STAR
T);
        break;
        case
DayOfWeek.Tuesday:

Notification.Comment(SCRIPT_ST
ART);
        break;
        case
DayOfWeek.Wednesday:
```

```
Notification.Print(SCRIPT_STAR  
T);  
        break;  
    case  
DayOfWeek.Thursday:  
  
    Notification.PlaySound("sound.  
wav");  
        break;  
    default:  
  
    Notification.Alert(SCRIPT_STAR  
T);  
        break;  
    }  
}  
  
public override  
void Update(TickStatus args)  
{  
}  
}  
}  
}
```



See Also

[Reference](#)

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptStatistic Property

Statistic entity

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public Statistic Statistic {  
    get; }
```

[Copy](#)

Property Value
[Statistic](#)

► See Also

Reference

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptTools

Property

Tool assistance in platform

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public Tools Tools { get; }
```

[Copy](#)

Property Value
[Tools](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
ToolsExample : StrategyBuilder
    {
        public
ToolsExample()
            : base()
        {

```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"ToolsExample";
        #endregion

base.SeparateWindow = false;
}

    public override
void Init()
{
    DateTime
dateTime =
Tools.TimeConvertor.CurrentTim
eZoneToUtc(DateTime.Now);

Notification.Print("UTC time
zone: " +
dateTime.ToString("yyyy-MM-dd HH:mm:ss"));
}

    public override
void Update(TickStatus args)
{
}

}
}
```

See Also

Reference

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScript Methods

The [BaseScript](#) type exposes the following members.

Methods

Name	Description
 Complete	This function is called before the script starts. (Overrides CSharpIndicator.Complete .)
 Dispose	(Inherited from CSharpScript)
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
 GetArray(ArrayList)	(Inherited from ScriptBase)
 GetArray(ArrayList, Boolean)	(Inherited from ScriptBase)
 GetArray(Type, Int32)	(Inherited from ScriptBase)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 Init	This function is called once before the script starts. (Overrides CSharpIndicator.Init .)
 LoadAndInvoke	(Inherited from CSharpScript)
 MemberwiseClone	Creates a shallow copy of the current instance. (Inherited from Object)
 OnDebug	(Inherited from CSharpScript)
 OnPaintChart	(Inherited from CSharpScript)

PushMethodDump	(Inherited from CSharpScript)
QueueUserWorkItem(WaitCallback)	(Inherited from CSharpScript)
QueueUserWorkItem(WaitCallback, Object)	(Inherited from CSharpScript)
QueueUserWorkItem(WaitCallback, Object, ApartmentState)	(Inherited from CSharpScript)
RegisterLocalVariable(String, FuncObject, String)	(Inherited from CSharpScript)
RegisterLocalVariable(String, FuncObject, ActionObject, String)	(Inherited from CSharpScript)
ToString	Returns a string that represents the current object. (Inherited from Object)
Update	This function is called amid new bar or history processing. argument with a tick status to quote state (args.TickStatus, IsHistory/IsBar) (Overrides CSharpIndicator.Update(TickEventArgs))

[Top](#)

See Also

[Reference](#)

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptComplete Method

This function is called before removing script

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override void  
Complete()
```

[Copy](#)

Implements
ICSharpIndicator.Complete

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
  
namespace TestEnv  
{  
    public class  
CompleteExample :  
IndicatorBuilder  
    {  
        public  
CompleteExample()  
        : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "CompleteExample";  
    #endregion  
  
    Lines.Set("CompleteExample");  
  
    base.SeparateWindow = false;  
}  
  
public override  
void Init()  
{  
  
}  
  
public override  
void Update(TickStatus args)  
{  
  
}  
  
public override  
void Complete()  
{  
  
    Notification.Print("Complete  
the script's work");  
}  
}
```

See Also

Reference

BaseScript Class
TradeApi.ToolBelt Namespace

BaseScriptInit Method

This function is called once before script's work starts

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public override void Init()
```

[Copy](#)

Implements
ICSharpIndicator.Init

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
    InitExample : IndicatorBuilder
    {
        public
        InitExample()
            : base()
        {
            #region
            Initialization
        }
    }
}
```

[Copy](#)

```
Credentials.ProjectName =
"InitExample";
#endregion

Lines.Set("InitExample");

base.SeparateWindow = false;
}

public override
void Init()
{
    Notification.Alert("Starting
the script's work");
}

public override
void Update(TickStatus args)
{
}

}

}
```

See Also

[Reference](#)

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptUpdate Method

This function is called amid new quote, new bar or history processing. It contains argument with a tick status to monitor quote state
(args.TickStatus,IsHistory/IsBar/IsQuote)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override void Update(Copy
    TickStatus args
)
```

Parameters

args **TickStatus**

TickStatus enum value either IsBar or IsQuote after history processing with initial IsHistory

Implements

ICSharpIndicator.Update(TickStatus)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;

namespace TestEnv
{
    public class
UpdateExample : 
IndicatorBuilder
    {
        public
UpdateExample()
            : base()
        {
            #region
Initialization

Credentials.ProjectName =
"UpdateExample";
            #endregion

Lines.Set("UpdateExample");

base.SeparateWindow = false;
        }

        public override
void Init()
        {

        }

        public override
void Update(TickStatus args)
        {
            if (args !=
TickStatus.IsHistory && args
== TickStatus.IsBar)
```

```
Notification.Print("Update on  
new bar");  
else  
  
Notification.Print("Update on  
new quote");  
}  
}  
}
```

See Also

[Reference](#)

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScript Fields

The [BaseScript](#) type exposes the following members.

↳ Fields

	Name	Description
	currentInternalInstrument	
	ScriptCache	(Inherited from CSharpScript)

[Top](#)

↳ See Also

[Reference](#)

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

BaseScriptcurrentInternalInstrument Field

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
protected [T:5zA=.8Ng=]  
currentInternalInstrument
```

[Copy](#)

Field Value
[T:5zA=.8Ng=]

► See Also

[Reference](#)

[BaseScript Class](#)

[TradeApi.ToolBelt Namespace](#)

Connection Class

Connection entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.ToolBeltConnection](#)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public sealed class Connection
```

[Copy](#)

The [Connection](#) type exposes the following members.

▪ Properties

	Name	Description
	ConnectionLossBehavior	Connection loss behavior
	Status	Retrieves current connection status

[Top](#)

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

▪ Events

Name	Description
------	-------------



OnConnectionChange Occurs when the connection state is changed
(`ConnectionState.Connected/Disconnected`)

[Top](#)

See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

Connection Properties

The [Connection](#) type exposes the following members.

► Properties

Name	Description
  ConnectionLossBehavior	Connection loss behavior
  Status	Retrieves current connection status

[Top](#)

► See Also

[Reference](#)

[Connection Class](#)

[TradeApi.ToolBelt Namespace](#)

ConnectionLossBehavior Property

Connection loss behavior

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public ConnectionLossBehavior  
ConnectionLossBehavior { get;  
set; }
```

[Copy](#)

Property Value
[ConnectionLossBehavior](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
ConnectionLossBehaviorExample  
: StrategyBuilder  
    {  
        public
```

[Copy](#)

```
ConnectionLossBehaviorExample()
    : base()
{
    #region Initialization

    Connection.ConnectionLossBehavior =
        ConnectionLossBehavior.Stop;
    #endregion
}

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
}

}
```

See Also

Reference

[Connection Class](#)

[TradeApi.ToolBelt Namespace](#)

ConnectionStatus Property

Retrieves current connection status

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public ConnectionStatus Status
{ get; }
```

[Copy](#)

Property Value
[ConnectionStatus](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
StatusExample :
StrategyBuilder
{
    public
StatusExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "StatusExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
}

ConnectionStatus status =
Connection.Status;

Notification.Print(status.ToString());
}
}
}
```

See Also

[Reference](#)

[Connection Class](#)

[TradeApi.ToolBelt Namespace](#)

Connection Methods

The [Connection](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[Connection Class](#)

[TradeApi.ToolBelt Namespace](#)

Connection Events

The [Connection](#) type exposes the following members.

▪ Events

	Name	Description
 	OnConnectionChange	Occurs when the connection state is changed (<code>ConnectionState.Connected/Disconnected</code>)

[Top](#)

▪ See Also

[Reference](#)

[Connection Class](#)

[TradeApi.ToolBelt Namespace](#)

ConnectionOnConnectionChangeEvent

Occurs when the connection state is changed
(`ConnectionState.Connected/Disconnected`)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event  
Action<ConnectionState>  
OnConnectionChange
```

[Copy](#)

Value

[ActionConnectionState](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
OnConnectionChangeExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
OnConnectionChangeExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"OnConnectionChangeExample";
        #endregion
    }

        public override
void Init()
    {

Connection.OnConnectionChange
+= connection_OnChange;
    }

        public void
connection_OnChange(ConnectionString
State state)
    {

Notification.Print(state.ToString());
    }

        public override
void Update(TickStatus args)
    {

}
}
}
```

See Also

Reference

[Connection Class](#)

[TradeApi.ToolBelt Namespace](#)

ConnectionLoss Behavior Enumeration

ConnectionLossBehavior types

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum  
ConnectionLossBehavior
```

[Copy](#)

► Members

Member name	Value	Description
Restart	0	Restart
Stop	1	Stop

► See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

ConnectionState Enumeration

ConnectionState type

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public enum ConnectionState
```

[Copy](#)

► Members

Member name	Value	Description
Success	0	Success
Fail	1	Fail

► See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

ConnectionStatus Enumeration

ConnectionStatus types

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum ConnectionStatus
```

[Copy](#)

► Members

Member name	Value	Description
Connected	0	Connected
Connecting	1	Connecting
Disconnected	2	Disconnected
ConnectionLost	3	ConnectionLost

► See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable Class

Global variable entry

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.ToolBeltGlobalVariable](#)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
GlobalVariable
```

[Copy](#)

The [GlobalVariable](#) type exposes the following members.

▪ Constructors

	Name	Description
	GlobalVariable	Initializes new variable

[Top](#)

▪ Properties

	Name	Description
		

Name	Variable name
Value	Variable value

[Top](#)

◀ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Concats name and value of the global variable (Overrides Object.ToString)

[Top](#)

See Also

Reference

[TradeApi.ToolBelt Namespace](#)

GlobalVariable Constructor

Initializes new variable

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public GlobalVariable(  
    string name,  
    Object value  
)
```

[Copy](#)

Parameters

name [String](#)
Variable name
value [Object](#)
Variable value

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{
```

[Copy](#)

```
    public class
ValueExample :
IndicatorBuilder
{
    public
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion

base.SeparateWindow = false;
}

    public override
void Init()
{
    if
(GlobalVariableManager.Count >
0)
{
    var
globalList =
GlobalVariableManager.GetGloba
lVariableList();
    var period
= 12;

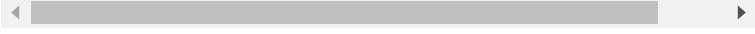
globalList.Add(
new
GlobalVariable("globalVariable
Period", period)
);
}
}
```

```
        }

    public override
void Update(TickStatus args)
{
    }

}

}
```



See Also

Reference

[GlobalVariable Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable Properties

The [GlobalVariable](#) type exposes the following members.

Properties

	Name	Description
 	Name	Variable name
 	Value	Variable value

[Top](#)

See Also

[Reference](#)

[GlobalVariable Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariableName Property

Variable name

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public string Name { get; }
```

[Copy](#)

Property Value
[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
NameExample : IndicatorBuilder
    {
        public
NameExample()
            : base()
        {

```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"NameExample";
        #endregion

base.SeparateWindow = false;
    }

        public override
void Init()
{
    if
(GlobalVariableManager.Count >
0)
    {
        var
globalList =
GlobalVariableManager.GetGloba
lVariableList();

globalList.ForEach(el =>
{
    Notification.Print(el.Name);
}
);
    }
}

        public override
void Update(TickStatus args)
{
}

    }
}
```

See Also

[Reference](#)

[GlobalVariable Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariableValue Property

Variable value

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public Object Value { get;  
    set; }
```

[Copy](#)

Property Value
[Object](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
ValueExample :  
IndicatorBuilder  
    {  
        public  
ValueExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ValueExample";
        #endregion

        base.SeparateWindow = false;
    }

    public override
void Init()
{
    if
(GlobalVariableManager.Count >
0)
{
    var
globalList =
GlobalVariableManager.GetGloba
lVariableList();

    globalList.ForEach(el =>
{
    Notification.Print(el.Value.To
String());
}
);
}

    public override
void Update(TickStatus args)
{
}
```

```
    }  
}
```



See Also

[Reference](#)

[GlobalVariable Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable Methods

The [GlobalVariable](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Concats name and value of the global variable

(Overrides
[ObjectToString](#))

[Top](#)

◀ See Also

[Reference](#)

[GlobalVariable Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariableToString Method

Concats name and value of the global variable

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

▪ Syntax

C#

```
public override string  
ToString()
```

[Copy](#)

Return Value
[String](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
ValueExample :  
IndicatorBuilder  
{  
    public
```

[Copy](#)

```
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion

base.SeparateWindow = false;
}

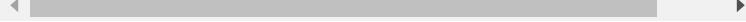
        public override
void Init()
{
    if
(GlobalVariableManager.Count >
0)
{
    var
globalList =
GlobalVariableManager.GetGloba
lVariableList();

globalList.ForEach(el =>
{
    Notification.Print(el.ToString
());
}
);

}
}

        public override
void Update(TickStatus args)
{
```

```
        }  
    }  
}
```



See Also

[Reference](#)

[GlobalVariable Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariableManager Class

GlobalVariableManager entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.ToolBeltGlobalVariableManager](#)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class
GlobalVariableManager
```

[Copy](#)

The [GlobalVariableManager](#) type exposes the following members.

► Constructors

	Name	Description
≡	GlobalVariableManager	Initializes a new instance of the GlobalVariableManager class

[Top](#)

► Properties

	Name	Description
≡ S F	Count	Returns variables' count in global storage

[Top](#)

Methods

Name	Description
   Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
   Exists	Check if variable with specified name exists in global storage
   Flush	Saves all serializable variables to disk
   GetGlobalVariableList	Returns all global variables as list
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current instance. (Inherited from Object)
   GetValue	Returns variable value by name
   Remove	Removes

		specified variable from global storage
  	RemoveAll	Removes all variables from global storage
  	SetValue	Removes all variables from global storage
 	ToString	Returns a string that represents the current object. (Inherited from Object)
  	TryGetValue	Performs a variable assigning from a global storage if such name exists in a scope

[Top](#)

See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable Manager Constructor

Initializes a new instance of the
[GlobalVariableManager](#) class

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public GlobalVariableManager() Copy
```

◀ See Also

[Reference](#)

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable Manager Properties

The [GlobalVariableManager](#) type exposes the following members.

Properties

	Name	Description
  	Count	Returns variables' count in global storage

[Top](#)

See Also

[Reference](#)

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable ManagerCount Property

Returns variables' count in global storage

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static int Count { get; } Copy
```

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
ValueExample : IndicatorBuilder
{
```

```
        public
ValueExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ValueExample";
    #endregion

base.SeparateWindow = false;
}

        public override
void Init()
{
    if
(GlobalVariableManager.Count >
0)
{
    var
globalList =
GlobalVariableManager.GetGloba
lVariableList();

if(globalList.Count > 0)

Notification.Print("Your
session obtains " +
GlobalVariableManager.Count.To
String() + " global
variables");
}

        public override
void Update(TickStatus args)
{
```

```
        }  
    }  
}
```



◀ See Also

Reference

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable Manager Methods

The [GlobalVariableManager](#) type exposes the following members.

Methods

Name	Description
   Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
   Exists	Check if variable with specified name exists in global storage
   Flush	Saves all serializable variables to disk
   GetGlobalVariableList	Returns all global

variables as
list



GetHashCode

Serves as
the default
hash
function.
(Inherited
from [Object](#))



GetType

Gets the
[Type](#) of the
current
instance.
(Inherited
from [Object](#))



GetValue

Returns
variable
value by
name



Remove

Removes
specified
variable
from global
storage



RemoveAll

Removes all
variables
from global
storage



SetValue

Removes all
variables
from global
storage



ToString

Returns a

string that represents the current object.
(Inherited from [Object](#))



[TryGetValue](#)

Performs a variable assigning from a global storage if such name exists in a scope

[Top](#)

◀ See Also

[Reference](#)

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable ManagerExists Method

Check if variable with specified name exists in global storage

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool Exists(Copy
                         string name
                         )
```

Parameters

name [String](#)

Variable name

Return Value

[Boolean](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.ToolBelt;

namespace TestEnvCopy
```

```
{  
    public class  
ValueExample :  
IndicatorBuilder  
    {  
        public  
ValueExample()  
            : base()  
        {  
            #region  
Initialization  
  
Credentials.ProjectName =  
"ValueExample";  
            #endregion  
  
base.SeparateWindow = false;  
        }  
  
        public override  
void Init()  
        {  
            object val = 15;  
            if  
(GlobalVariableManager.Count >  
0 &&  
GlobalVariableManager.Exists("globalVariablePeriod"))  
            {  
  
Notification.Print("Your  
session has this global  
variable");  
            }  
            else  
  
GlobalVariableManager.SetValue  
("globalVariablePeriod", val,  
VariableLifetime.SaveSession);
```

```
        }

    public override
void Update(TickStatus args)
{
    }

}

}
```



See Also

Reference

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable ManagerFlush Method

Saves all serializable variables to disk

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static void Flush()
```

[Copy](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.ToolBelt;
```

[Copy](#)

```
namespace TestEnv
{
    public class
FlushExample :
IndicatorBuilder
{
    public
FlushExample()
    : base()
    {
        #region
Initialization
    }
}
```

```
Credentials.ProjectName =
"FlushExample";
#endregion

base.SeparateWindow = false;
}

int period = 0;

public override
void Init()
{
    GlobalVariableManager.SetValue
("globalVariablePeriod",
period,
VariableLifetime.SaveFile);

GlobalVariableManager.Flush();
}

public override
void Update(TickStatus args)
{
    }
}
```



See Also

[Reference](#)

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariableManager.GetGlobalVariableList Method

Returns all global variables as list

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static  
List<GlobalVariable>  
GetGlobalVariableList()
```

[Copy](#)

Return Value

[ListGlobalVariable](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
GetGlobalVariableListExample :  
IndicatorBuilder
```

[Copy](#)

```
        {
            public
GetGlobalVariableListExample()
            : base()
            {
                #region
Initialization

Credentials.ProjectName =
"GetGlobalVariableListExample"
;

                #endregion

base.SeparateWindow = false;
            }
        }

            public override
void Init()
            {
                if
(GlobalVariableManager.Count >
0)
                {
                    var
globalList =
GlobalVariableManager.GetGloba
lVariableList();

globalList.ForEach(el =>
{
    Notification.Print(el.Name);
}
);

}
            }

            public override
void Update(TickStatus args)
```

```
{  
}  
}  
}
```



◀ See Also

Reference

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable ManagerGetValue Method

Returns variable value by name

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public static Object GetValue(Copy
                           string name
                           )
```

Parameters

name [String](#)
Variable name

Return Value
[Object](#)

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using TradeApi.ToolBelt;
```

```
namespace TestEnv
{
    public class GetGlobalVariableListExample : IndicatorBuilder
    {
        public GetGlobalVariableListExample()
            : base()
        {
            #region Initialization

            Credentials.ProjectName =
                "GetGlobalVariableListExample"
            ;
            #endregion

            base.SeparateWindow = false;
        }

        int period = 0;

        public override void Init()
        {

            if(GlobalVariableManager.Exists("globalVariablePeriod"))
            {
                period =
                    (int)GlobalVariableManager.GetValue("globalVariablePeriod");
            }
        }

        public override void Update(TickStatus args)
        {
```

```
        }  
    }  
}
```



◀ See Also

Reference

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable ManagerRemove Method

Removes specified variable from global storage

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool Remove(Copy
    string name
)
```

Parameters

name [String](#)

Variable name

Return Value

[Boolean](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Indicators;
using TradeApi.ToolBelt;
```

```
namespace TestEnv
{
    public class GetGlobalVariableListExample : IndicatorBuilder
    {
        public GetGlobalVariableListExample() : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "GetGlobalVariableListExample";
            ;
            #endregion
        }

        public override void Init()
        {
            if(GlobalVariableManager.Exists("globalVariablePeriod"))
            {
                GlobalVariableManager.Remove("globalVariablePeriod");
            }
        }

        public override void Update(TickStatus args)
        {
```

```
        }  
    }  
}
```



◀ See Also

Reference

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariableManager.RemoveAll Method

Removes all variables from global storage

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static void RemoveAll() Copy
```

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Indicators;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
GetGlobalVariableListExample : IndicatorBuilder
    {
        public
GetGlobalVariableListExample()
            : base()
        {
```

```
        #region
Initialization

Credentials.ProjectName =
"GetGlobalVariableListExample"
;
#endregion

base.SeparateWindow = false;
}

public override
void Init()
{
    GlobalVariableManager.RemoveAll();

    if(GlobalVariableManager.Count
== 0)

Notification.Print("Your
session does not have any
global variables");
}

public override
void Update(TickStatus args)
{
}

}

}
```

◀ See Also
Reference

GlobalVariableManager Class
TradeApi.ToolBelt Namespace

GlobalVariable ManagerSetValue Method

Removes all variables from global storage

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static void SetValue(Copy
    string name,
    Object value,
    VariableLifetime
    lifetime
)
```

Parameters

name [String](#)

Variable name

value [Object](#)

New value

lifetime [VariableLifetime](#)

Varialbe LifeTime

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
GetGlobalVariableListExample : 
IndicatorBuilder
    {
        public
GetGlobalVariableListExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"GetGlobalVariableListExample"
;
            #endregion

base.SeparateWindow = false;
        }

        int period = 0;

        public override
void Init()
        {

GlobalVariableManager.SetValue
("globalVariablePeriod",
period,
VariableLifetime.SaveSession);

GlobalVariableManager.SetValue
("globalVariablePeriod" +
period +
```

```
Credentials.ProjectName,  
period,  
VariableLifetime.SaveSession);  
}  
  
        public override  
void Update(TickStatus args)  
{  
  
}  
}  
}
```



See Also

Reference

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

GlobalVariable ManagerTryGetValue Method

Performs a variable assigning from a global storage if such name exists in a scope

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool
TryGetValue(
    string name,
    out Object obj
)
```

[Copy](#)

Parameters

name [String](#)

Variable name

obj [Object](#)

Variable value

Return Value

[Boolean](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Indicators;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
TryGetValueExample : 
IndicatorBuilder
    {
        public
TryGetValueExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"TryGetValueExample";
            #endregion

base.SeparateWindow = false;
        }

        public override
void Init()
        {
            if
(GlobalVariableManager.TryGetValue(
    "globalVariablePeriod",
    out object new_period))

Notification.Print("New
variable is assigned from
globals: " +
new_period.ToString());
        }
}
```

```
        public override
void Update(TickStatus args)
{
    }

}
```

See Also

Reference

[GlobalVariableManager Class](#)

[TradeApi.ToolBelt Namespace](#)

Notification Class

Notification batch entity

▪ Inheritance Hierarchy

[SystemObject](#) `TradeApi.ToolBeltNotification`

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
Notification
```

[Copy](#)

The `Notification` type exposes the following members.

▪ Methods

	Name	Description
	Alert	Shows all given args in message screen
	Comment	Prints data within chart window

	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	PlaySound	Plays one of the standard sounds
	Print	Prints out the args in logs
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

See Also

[Reference](#)

TradeApi.ToolBelt Namespace

Notification Methods

The [Notification](#) type exposes the following members.

▪ Methods

	Name	Description
 	Alert	Shows all given args in message screen
 	Comment	Prints data within chart window
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[PlaySound](#)

Plays one of
the standard
sounds



[Print](#)

Prints out the
args in logs



[ToString](#)

Returns a
string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

Reference

[Notification Class](#)

[TradeApi.ToolBelt Namespace](#)

NotificationAlert Method

Shows all given args in message screen

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void Alert(  
    string message  
)
```

[Copy](#)

Parameters

message String
args to notify

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
AlertExample : StrategyBuilder  
    {
```

[Copy](#)

```
        public
AlertExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"AlertExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
positions =
PositionsManager.GetPositions(
position =>
position.Instrument.Symbol ==
InstrumentsManager.Current.Sym
bol);

Notification.Alert($"Position'
s count: {positions.Count}");
}
}
}
```

See Also

Reference

[Notification Class](#)

[TradeApi.ToolBelt Namespace](#)

NotificationComment Method

Prints data within chart window

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public void Comment(  
                      string message  
)
```

[Copy](#)

Parameters

message String
args to notify

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
CommentExample :  
StrategyBuilder
```

[Copy](#)

```
        {
            public
CommentExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"CommentExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
position =>
position.Instrument.Symbol ==
InstrumentsManager.Current.Sym
bol);

Notification.Comment($"Positio
n's count:
{positions.Count}");
    }
}
}
```

See Also

[Reference](#)

[Notification Class](#)

[TradeApi.ToolBelt Namespace](#)

NotificationPlaySound Method

Plays one of the standard sounds

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void PlaySound(  
    string filePath  
)
```

[Copy](#)

Parameters

filePath [String](#)
path to sound file

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
using System;  
using System.IO;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
PlaySoundExample :  
StrategyBuilder  
{  
    public  
PlaySoundExample()  
        : base()  
{  
    #region  
Initialization  
  
Credentials.ProjectName =  
"PlaySoundExample";  
    #endregion  
}  
  
    public override  
void Init()  
{  
}  
  
    public override  
void Update(TickStatus args)  
{  
    if (args ==  
TickStatus.IsBar)  
    {  
        var  
positions =  
PositionsManager.GetPositions(  
position =>  
position.Instrument.Symbol ==  
InstrumentsManager.Current.Sym  
bol);  
  
if (positions.Count>0)  
  
Notification.PlaySound(Path.Co  
mbine(Environment.GetFolderPath(Environment.SpecialFolder.Co  
mmonMusic), "play.wav"));
```

```
        }  
    }  
}
```

See Also

[Reference](#)

[Notification Class](#)

[TradeApi.ToolBelt Namespace](#)

NotificationPrint Method

Prints out the args in logs

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void Print(  
    string message  
)
```

[Copy](#)

Parameters

message String
args to notify

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.History;  
  
namespace TestEnv  
{  
    public class  
PrintExample : StrategyBuilder  
    {
```

[Copy](#)

```
        public
PrintExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"PrintExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
positions =
PositionsManager.GetPositions(
position =>
position.Instrument.Symbol ==
InstrumentsManager.Current.Sym
bol);

Notification.Print($"Position'
s count: {positions.Count}");
}
}
}
```

See Also

Reference

[Notification Class](#)

[TradeApi.ToolBelt Namespace](#)

ScreenCapture Class

ScreenCapture entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.ToolBeltScreenCapture](#)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
    ScreenCapture
```

[Copy](#)

The [ScreenCapture](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)

	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
 	MakeScreenShot	Returns image object with screenshot
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

ScreenCapture Methods

The [ScreenCapture](#) type exposes the following members.

Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	MakeScreenShot	Returns

image
object with
screenshot



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[ScreenCapture Class](#)

[TradeApi.ToolBelt Namespace](#)

ScreenCaptureMake ScreenShot Method

Returns image object with screenshot

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public Image MakeScreenShot(  
    Rectangle rectangle,  
    Size maxSize  
)
```

[Copy](#)

Parameters

rectangle [Rectangle](#)

rectangle to take a screen shot

maxSize [Size](#)

max size to take a screen shot

Return Value

[Image](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.History;  
using System.Drawing;
```

[Copy](#)

```
using TradeApi.Indicators;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
MakeScreenShotExample : 
IndicatorBuilder
{
    public
MakeScreenShotExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"MakeScreenShotExample";
        #endregion
    }

        ScreenCapture
scrCapt;
        Rectangle? rect;
        BuiltInIndicator
ema;

        public override
void Init()
    {
        scrCapt =
Tools.ScreenCapture;
        rect =
this.ChartSource.GetAllWindows
().Find(window =>
window.WindowNumber ==
this.ChartSource.FindWindow(th
is))?.WindowRectangle;
        ema =
IndicatorsManager.BuildIn.MA (H
```

```
istoryDataSeries, 200,
MAMode.EMA);
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        if
(ema.GetValue()
<HistoryDataSeries.GetValue(Pr
iceType.Close, 1))
        {
            if
(rect.Value != null)

scrCapt.MakeScreenShot(rect.Va
lue, new
Size(rect.Value.Width,
rect.Value.Height));
        }
    }
}

```



See Also

Reference

[ScreenCapture Class](#)

[TradeApi.ToolBelt Namespace](#)

Statistic Class

Statistic entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.ToolBeltStatistic](#)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class Statistic
```

[Copy](#)

The `Statistic` type exposes the following members.

► Properties

	Name	Description
	AcceptedOrders	Gets the count of accepted orders
	Balance	Gets balance
	CancelledOrders	Gets the count of

		cancelled orders
	FilledOrders	Gets the count of filled orders
	LongPos	Gets the count of long positions
	OpenGrossPnL	Gets the open gross PnL
	RejectedOrders	Gets the count of rejected orders
	SendedOrders	Gets the count of submitted orders
	ShortPos	Gets the count of short positions
	StartTimeUtc	Get start time in UTC
	TotalGrossPnL	Gets the total gross PnL
	WorkTime	Get work time

[Top](#)

◀ Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

Statistic Properties

The [Statistic](#) type exposes the following members.

Properties

	Name	Description
 	AcceptedOrders	Gets the count of accepted orders
 	Balance	Gets balance
 	CancelledOrders	Gets the count of cancelled orders
 	FilledOrders	Gets the count of filled orders
 	LongPos	Gets the count of long positions
 	OpenGrossPnL	Gets the open gross PnL
 	RejectedOrders	Gets the count of

		rejected orders
 	SendedOrders	Gets the count of submitted orders
 	ShortPos	Gets the count of short positions
 	StartTimeUtc	Get start time in UTC
 	TotalGrossPnL	Gets the total gross PnL
 	WorkTime	Get work time

[Top](#)

◀ See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticAcceptedOrders Property

Gets the count of accepted orders

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int AcceptedOrders {  
    get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
AcceptedOrdersExample :  
StrategyBuilder  
    {  
        public  
AcceptedOrdersExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "AcceptedOrders";
        #endregion
    }

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.A
cceptedOrders.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticBalance Property

Gets balance

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Balance { get; } Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
BalanceExample :
StrategyBuilder
{
    public
BalanceExample()
    : base()
```

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "Balance";  
    #endregion  
}  
  
    public override  
    void Update(TickStatus args)  
    {  
  
        Notification.Print(Statistic.B  
        alance.ToString());  
    }  
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticCancelledOrders Property

Gets the count of cancelled orders

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int CancelledOrders {  
    get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
CancelledOrdersExample :  
StrategyBuilder  
    {  
        public  
CancelledOrdersExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "CancelledOrders";
        #endregion
    }

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.C
ancelledOrders.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticFilledOrders Property

Gets the count of filled orders

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int FilledOrders { get;  
}
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
FilledOrdersExample :  
StrategyBuilder  
    {  
        public  
FilledOrdersExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "FilledOrders";
        #endregion
    }

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.F
illedOrders.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticLongPos Property

Gets the count of long positions

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public int LongPos { get; }
```

[Copy](#)

Property Value
[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
LongPosExample :
StrategyBuilder
{
    public
LongPosExample()
    : base()
```

[Copy](#)

```
    {
        #region
Initialization

Credentials.ProjectName =
"LongPos";
        #endregion
    }

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.L
ongPos.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticOpenGrossPnL Property

Gets the open gross PnL

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double OpenGrossPnL {  
    get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
        OpenGrossPnLExample :  
            StrategyBuilder  
    {  
        public  
            OpenGrossPnLExample()
```

[Copy](#)

```
: base()
{
    #region
Initialization

Credentials.ProjectName =
"OpenGrossPnL";
    #endregion
}

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.O
penGrossPnL.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticRejectedOrders Property

Gets the count of rejected orders

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int RejectedOrders {  
    get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
RejectedOrdersExample :  
StrategyBuilder  
    {  
        public  
RejectedOrdersExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "RejectedOrders";
        #endregion
    }

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.R
ejectedOrders.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticSendedOrders Property

Gets the count of submitted orders

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public int SendedOrders { get;  
}
```

[Copy](#)

Property Value
[Int32](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
SendedOrdersExample :  
StrategyBuilder  
{  
    public  
SendedOrdersExample()
```

[Copy](#)

```
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"SendedOrders";
        #endregion
    }

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.S
endedOrders.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticShortPos Property

Gets the count of short positions

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public int ShortPos { get; } Copy
```

Property Value
[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using TradeApi.ToolBelt;

namespace TestEnv
{
    public class
ShortPosExample :
StrategyBuilder
{
    public
ShortPosExample()
        : base()
```

```
    {
        #region
Initialization

Credentials.ProjectName =
"ShortPos";
        #endregion
    }

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.S
hortPos.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticStartTimeUtc Property

Get start time in UTC

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public DateTime StartTimeUtc {  
    get; }
```

[Copy](#)

Property Value
[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
StartTimeExample :  
StrategyBuilder  
{  
    public  
StartTimeExample()
```

[Copy](#)

```
: base()
{
    #region
Initialization

Credentials.ProjectName =
"StartTime";
    #endregion
}

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.S
tartTime);
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticTotalGrossPnL Property

Gets the total gross PnL

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double TotalGrossPnL {  
    get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
TotalGrossPnLExample :  
StrategyBuilder  
    {  
        public  
TotalGrossPnLExample()
```

[Copy](#)

```
: base()
{
    #region
Initialization

Credentials.ProjectName =
"TotalGrossPnL";
    #endregion
}

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.T
otalGrossPnL.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

StatisticWorkTime Property

Get work time

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public TimeSpan WorkTime {  
    get; }
```

[Copy](#)

Property Value
[TimeSpan](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using TradeApi.ToolBelt;  
  
namespace TestEnv  
{  
    public class  
WorkTimeExample :  
StrategyBuilder  
    {  
        public  
WorkTimeExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "WorkTime";
        #endregion
    }

        public override
void Update(TickStatus args)
{
    Notification.Print(Statistic.W
orkTime.ToString());
}
}
```

See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

Statistic Methods

The [Statistic](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[Statistic Class](#)

[TradeApi.ToolBelt Namespace](#)

TimeConvertor Class

TimeConvertor entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.ToolBeltTimeConvertor](#)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
TimeConvertor
```

[Copy](#)

The [TimeConvertor](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)

 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	ToString	Returns a string that represents the current object. (Inherited from Object)
 	UtcToCurrentTimeZone	Converts UTC time to selected time zone
 	CurrentTimeZoneToUtc	Converts selected time zone to UTC time

[Top](#)

See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

TimeConvertor Methods

The [TimeConvertor](#) type exposes the following members.

▲ Methods

Name	Description
Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode	Serves as the default hash function. (Inherited from Object)
GetType	Gets the Type of the current instance. (Inherited from Object)
ToString	Returns a

string that represents the current object.
(Inherited from [Object](#))

 [UtcToCurrentTimeZone](#) Converts UTC time to selected time zone

 [CurrentTimeZoneToUtc](#) Converts selected time zone to UTC time

[Top](#)

See Also

[Reference](#)

[TimeConvertor Class](#)

[TradeApi.ToolBelt Namespace](#)

TimeConvertorUtcTo CurrentTimeZone Method

Converts UTC time to selected time zone

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public DateTime  
UtcToCurrentTimeZone(  
    DateTime timeUtc  
)
```

[Copy](#)

Parameters

timeUtc [DateTime](#)

DateTime in UTC to perform conversion
to current

Return Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;
```

[Copy](#)

```
using System;

namespace TestEnv
{
    public class
UtcToCurrentTimeZoneExample :  
StrategyBuilder
{
    public
UtcToCurrentTimeZoneExample()
: base()
{
    #region
Initialization

Credentials.ProjectName =
"UtcToCurrentTimeZoneExample";
    #endregion
}

    public override
void Init()
{
    DateTime
dateTime =
Tools.TimeConvertor.UtcToCurre  
ntTimeZone(DateTime.UtcNow);

Notification.Print("Current
time zone: " +
dateTime.ToString("yyyy-MM-dd HH:mm:ss"));
}

    public override
void Update(TickStatus args)
{
}
}
```

See Also

Reference

[TimeConvertor Class](#)

[TradeApi.ToolBelt Namespace](#)

TimeConvertorCurrent TimeZoneToUtc Method

Converts selected time zone to UTC time

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public DateTime  
CurrentTimeZoneToUtc(  
    DateTime time  
)
```

[Copy](#)

Parameters

time [DateTime](#)

DateTime to perform conversion to UTC

Return Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System;
```

[Copy](#)

```
namespace TestEnv
{
    public class CurrentTimeZoneToUtcExample : StrategyBuilder
    {
        public CurrentTimeZoneToUtcExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "CurrentTimeZoneToUtcExample";
            #endregion
        }

        public override void Init()
        {
            DateTime
            date = Tools.TimeConvertor.CurrentTime
            zoneToUtc(DateTime.UtcNow);

            Notification.Print("UTC time
zone: " +
date.ToString("yyyy-MM-dd HH:mm:ss"));
        }

        public override void Update(TickStatus args)
        {
        }
    }
}
```

See Also

[Reference](#)

[TimeConvertor Class](#)

[TradeApi.ToolBelt Namespace](#)

Tools Class

Tools entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.ToolBeltTools](#)

Namespace: [TradeApi.ToolBelt](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class Tools
```

[Copy](#)

The [Tools](#) type exposes the following members.

► Properties

	Name	Description
	ScreenCapture	Screen capture tool
	TimeConverter	Time converter tool

[Top](#)

► Methods

	Name	Description
--	------	-------------

≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetType	Gets the Type of the current instance. (Inherited from Object)
≡	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

Tools Properties

The [Tools](#) type exposes the following members.

► Properties

	Name	Description
	ScreenCapture	Screen capture tool
	TimeConvertor	Time converter tool

[Top](#)

► See Also

[Reference](#)

[Tools Class](#)

[TradeApi.ToolBelt Namespace](#)

ToolsScreenCapture Property

Screen capture tool

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public ScreenCapture  
ScreenCapture { get; }
```

[Copy](#)

Property Value
[ScreenCapture](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Indicators;  
using TradeApi.ToolBelt;  
using System.Drawing;  
using TradeApi.History;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
ScreenCaptureExample :  
IndicatorBuilder  
{
```

```
        public
ScreenCaptureExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"ScreenCaptureExample";
    #endregion
}

    ScreenCapture
scrCapt;
    Rectangle? rect;
    BuiltInIndicator
ema;

        public override
void Init()
{
    scrCapt =
Tools.ScreenCapture;
    rect =
this.ChartSource.GetAllWindows
().Find(window =>
window.WindowNumber ==
this.ChartSource.FindWindow(th
is))?.WindowRectangle;
    ema =
IndicatorsManager.BuildIn.MA(H
istoryDataSeries, 200,
MAMode.EMA);
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
```

```
        {
            if
(ema.GetValue()
<HistoryDataSeries.GetValue(Pr
iceType.Close, 1))
{
    if
(rect.Value != null)

scrCapt.MakeScreenShot(rect.Va
lue, new
Size(rect.Value.Width,
rect.Value.Height));
}
}
}
}
```



See Also

Reference

[Tools Class](#)

[TradeApi.ToolBelt Namespace](#)

ToolsTimeConverter Property

Time converter tool

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public TimeConvertor  
TimeConvertor { get; }
```

[Copy](#)

Property Value
[TimeConvertor](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System;  
  
namespace TestEnv  
{  
    public class  
    CurrentTimeZoneToUtcExample :  
    StrategyBuilder  
    {  
        public  
        CurrentTimeZoneToUtcExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "CurrentTimeZoneToUtcExample";
        #endregion
    }

        public override
void Init()
{
    DateTime
dateTime =
Tools.TimeConvertor.CurrentTim
eZoneToUtc(DateTime.Now);

Notification.Print("UTC time
zone: " +
dateTime.ToString("yyyy-MM-dd HH:mm:ss"));
}

        public override
void Update(TickStatus args)
{
}

}
}
```

See Also

[Reference](#)

[Tools Class](#)

[TradeApi.ToolBelt Namespace](#)

Tools Methods

The [Tools](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[Tools Class](#)

[TradeApi.ToolBelt Namespace](#)

VariableLifetime Enumeration

VariableLifetime types

Namespace: [TradeApi.ToolBelt](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public enum VariableLifetime
```

[Copy](#)

► Members

Member name	Value	Description
SaveSession	0	The variable will be removed on exit
SaveFile	1	The variable will be written to file

► See Also

[Reference](#)

[TradeApi.ToolBelt Namespace](#)

TradeApi.Trading Namespace

↳ Classes

Class	Description
 Account	Account entity
 AccountManager	AccountManager entity
 Order	Order entity
 OrderRequest	OrderRequest entity
 OrdersManager	OrdersManager entity
 OrdersViewer	OrdersViewer entity
 Position	Position entity
 PositionsManager	PositionsManager entity
 PositionsViewer	PositionsViewer entity
 ReplaceOrderRequest	ModifyOrderRequest entity
 StrategyBuilder	StrategyBuilder entity



TradeResult

TradeResult entity

▪ Enumerations

	Enumeration	Description
	AccountStatus	Account status type
	OrderSide	Order side type
	OrderStatus	Order's Status
	OrderType	Order's type
	PositionSide	Position side mode
	ProductType	Product Type
	SLTPPriceType	SI/Tp price type
	TimeInForce	TimeInForce

Account Class

Account entity

▪ Inheritance Hierarchy

[SystemObject](#) `TradeApi.TradingAccount`

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public sealed class Account
```

[Copy](#)

The `Account` type exposes the following members.

▪ Properties

	Name	Description
	<code>Id</code>	Account's Id
	<code>Name</code>	Account's Name
	<code>Status</code>	Account's current status

[Top](#)

▪ Methods

Name	Description
------	-------------

 Equals	Allow to compare two accounts (Overrides Object.Equals(Object))
  GetAccountDetails	Account's unique details available from server/vendor
 GetHashCode	(Overrides Object.GetHashCode)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

▲ Operators

Name	Description
  Equality(Account, Account)	Allow to compare two accounts
  Inequality(Account, Account)	Allow to compare two accounts

[Top](#)

◀ See Also

Reference

[TradeApi.Trading Namespace](#)

Account Properties

The [Account](#) type exposes the following members.

► Properties

	Name	Description
 	Id	Account's Id
 	Name	Account's Name
 	Status	Account's current status

[Top](#)

► See Also

[Reference](#)

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountId Property

Account's Id

Namespace: TradeApi.Trading

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Id { get; }
```

[Copy](#)

Property Value

String

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class IdExample
        : StrategyBuilder
    {
        public IdExample()
            : base()
        {
            #region
            Initialization
            Credentials.ProjectName =
        }
    }
}
```

[Copy](#)

```
"IdExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(SLTPin
clude: true);
        if
(orders.Count > 0)

orders.ForEach(order =>
{
    Notification.Print($"Account
ID is {order.Account.Id}");
} );
    }
}
}
```

See Also

[Reference](#)

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountName Property

Account's Name

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public string Name { get; }
```

[Copy](#)

Property Value
[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
NameExample : StrategyBuilder
    {
        public
NameExample()
            : base()
        {
            #region
```

[Copy](#)

```
Initialization

Credentials.ProjectName =
"NameExample";
                                #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(SLTPin
clude: true);
        if
(orders.Count > 0)

orders.ForEach(order =>
{
    Notification.Print($"Account
Name is
{order.Account.Name}");
} );
    }
}
}
```

See Also

Reference

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountStatus Property

Account's current status

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public AccountStatus Status {  
    get; }
```

[Copy](#)

Property Value

[AccountStatus](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
StatusExample :  
StrategyBuilder  
    {  
        public  
StatusExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"StatusExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(SLTPin
clude: true);
        if
(orders.Count > 0)

orders.ForEach(order =>
{
    if
(order.Account.Status ==
AccountStatus.Active)

Notification.Print($"Current
active account name is
{order.Account.Name}");
})
}
}
```

```
        }  
    }
```

See Also

[Reference](#)

[Account Class](#)

[TradeApi.Trading Namespace](#)

Account Methods

The [Account](#) type exposes the following members.

► Methods

	Name	Description
	Equals	Allow to compare two accounts (Overrides Object.Equals(Object))
	GetAccountDetails	Account's unique details available from server/vendor
	GetHashCode	(Overrides Object.GetHashCode)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

► See Also

Reference

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountEquals Method

Allow to compare two accounts

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override bool Equals(Copy
                           Object obj
                           )
```

Parameters

obj [Object](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountGetAccountDetails Method

Account's unique details available from server/vendor

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public Dictionary<string,  
    string> GetAccountDetails(  
        string assetName =  
        null  
    )
```

[Copy](#)

Parameters

assetName [String](#) (Optional)

Return Value

[DictionaryString, String](#)

► Example

C#

```
using  
System.Collections.Generic;  
using Runtime.Script;
```

[Copy](#)

```
using TradeApi.Trading;

namespace TestEnv
{
    public class
AccountDetailssExample :  
StrategyBuilder
    {
        public
AccountDetailssExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"AccountDetailssExample";
            #endregion
        }

        public override
void Init()
        {

        }

        public override
void Update(TickStatus args)
        {
            if (args ==
TickStatus.IsBar)
            {
                var orders
=
OrdersManager.GetOrders(SLTPin
clude: true);
                if
(orders.Count > 0)

orders.ForEach(order =>
```

```
        {
            if
                (order.Account.Status ==
                AccountStatus.Active)

            foreach (KeyValuePair<string,
                string> entry in
                order.Account.GetAccountDetail
                s())
            {

                Notification.Print($"AccountDe
                tails {entry.Key} :
                {entry.Value}");
            }
        }
    }
}
```

See Also

Reference

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountGetHashCode Method

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public override int  
GetHashCode()
```

[Copy](#)

Return Value

[Int32](#)

◀ See Also

[Reference](#)

[Account Class](#)

[TradeApi.Trading Namespace](#)

Account Operators

The [Account](#) type exposes the following members.

Operators

Name	Description
 S Equality(Account, Account)	Allow to compare two accounts
 S Inequality(Account, Account)	Allow to compare two accounts

[Top](#)

See Also

[Reference](#)

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountEquality Operator

Allow to compare two accounts

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public static bool operator ==  
(  
    Account acc1,  
    Account acc2  
)
```

[Copy](#)

Parameters

acc1 [Account](#)
acc2 [Account](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountInequality Operator

Allow to compare two accounts

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator !=  
(  
    Account acc1,  
    Account acc2  
)
```

[Copy](#)

Parameters

acc1 [Account](#)
acc2 [Account](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Account Class](#)

[TradeApi.Trading Namespace](#)

AccountManager Class

AccountManager entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingAccountManager](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version:

1.0.0

► Syntax

C#

```
public sealed class AccountManager
```

[Copy](#)

The `AccountManager` type exposes the following members.

► Properties

	Name	Description
	Current	Current account that is used by user in the trading session

[Top](#)

► Methods

	Name	Description
	Equals	Determines whether the specified

object is equal to the current object.
(Inherited from [Object](#))

 GetAccounts	Get all possible accounts in the trading session
 GetAccounts(PredicateAccount)	Get matched accounts in the trading session
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

↳ See Also

Reference

[TradeApi.Trading Namespace](#)

AccountManager Properties

The [AccountManager](#) type exposes the following members.

Properties

	Name	Description
 	Current	Current account that is used by user in the trading session

[Top](#)

See Also

[Reference](#)

[AccountManager Class](#)

[TradeApi.Trading Namespace](#)

AccountManagerCurrentProperty

Current account that is used by user in the trading session

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Account Current { get; } Copy
```

Property Value
[Account](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class CurrentExample :
        StrategyBuilder
    {
Copy
```

```
        public
CurrentExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"CurrentExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var account =
AccountManager.Current;
        var newRequest
= new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
account, OrderSide.Buy, 1);

OrdersManager.Send(newRequest)
;
    }
}
}
```

See Also

[Reference](#)

[AccountManager Class](#)

[TradeApi.Trading Namespace](#)

AccountManager Methods

The [AccountManager](#) type exposes the following members.

Methods

Name	Description
  Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
  GetAccounts	Get all possible accounts in the trading session
  GetAccounts(PredicateAccount)	Get matched accounts in the trading session
  GetHashCode	Serves as the default hash function. (Inherited from Object)



GetType

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[AccountManager Class](#)

[TradeApi.Trading Namespace](#)

AccountManagerGetAccounts Method

Overload List

Name	Description
  GetAccounts	Get all possible accounts in the trading session
  GetAccounts(PredicateAccount)	Get matched accounts in the trading session

[Top](#)

See Also

[Reference](#)

[AccountManager Class](#)

[TradeApi.Trading Namespace](#)

AccountManagerGet Accounts Method

Get all possible accounts in the trading session

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public List<Account>
GetAccounts()
```

[Copy](#)

Return Value

[ListAccount](#)

▪ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
GetAccountsExample :
StrategyBuilder
{
    public
GetAccountsExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "GetAccountsExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var account =
AccountManager.GetAccounts();

        account.ForEach(x =>
Notification.Comment(x.Name));
    }
}
}
```

See Also

Reference

[AccountManager Class](#)

[GetAccounts Overload](#)

[TradeApi.Trading Namespace](#)

AccountManagerGetAccounts(PredicateAccount) Method

Get matched accounts in the trading session

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public List<Account>
GetAccounts(
    Predicate<Account>
predicate
)
```

[Copy](#)

Parameters

predicate [PredicateAccount](#)

Method to be triggered as predicate delegate

Return Value

[ListAccount](#)

▪ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
```

[Copy](#)

```
namespace TestEnv
{
    public class
GetAccountsExample :
StrategyBuilder
{
    public
GetAccountsExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetAccountsExample";
        #endregion
    }

    public override void
Init()
{
}

    public override void
Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var account =
AccountManager.GetAccounts(x =>
x.Status ==
AccountStatus.Active);

account.ForEach(x =>
Notification.Comment(x.Name));
    }
}
}
```

See Also

[Reference](#)

[AccountManager Class](#)

[GetAccounts Overload](#)

[TradeApi.Trading Namespace](#)

AccountStatus Enumeration

Account status type

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public enum AccountStatus
```

[Copy](#)

► Members

Member name	Value	Description
Active	0	Active
Closed	1	Closed
Suspend	2	Suspend
TradingDisabledByRiskRules	3	TradingDisabledByRiskRules

► See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

Order Class

Order entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingOrder](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public sealed class Order :  
IDisposable
```

[Copy](#)

The [Order](#) type exposes the following members.

► Properties

Name	Description
 Account	Account that has invoked an order.
 AverageFilledPrice	An average price of the order fill
 BoundTo	Obsolete. The if of

		bound order. It is related to OCO
 	BoundToId	If it is a bound order.
 	Comment	Returns notation for the selected order
 	ExpirationTimeUtc	The date of expiration order in UTC for GTD orders
 	FilledQuantity	Filled quantity
 	ID	ID of an order
 	Instrument	Represents the information of instrument for order
 	IsActive	Checks if order is pending on server
 	IsAutoTrade	Checks if order is invoked by strategy

	IsStopLossOrder	Returns true if specified order is a slave Stop Loss order in context of primary order
	IsTakeProfitOrder	Returns true if specified order is a slave Take Profit order in context of primary order
	LastUpdateTimeUtc	Returns last modification time for the order in UTC
	Name	Custom name of an order given by user
	PlacedTimeUtc	Date and time when the order was placed in UTC
	Price	Price at which order can be executed
	ProductType	Order Product Type
	Quantity	Custom name of an order

		given by user
	Side	Side of the order (Buy or Sell)
	SLTPPriceType	Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute/tick or point
	Status	Checks order status
	StopLossLimitPrice	Returns a stop-loss limit price or null if none is in context of primary order
	StopLossPrice	Returns a stop-loss price or null if none is in context of primary order
	StopPrice	Stop Price of the Limit order
	TakeProfitPrice	Returns a take-profit

		price based on SLTPPriceType or null if none is in context of primary order
 TIF		Order's Time-in-force
 TrailingStopOffset		Returns a trailing offset value or null if none is in context of primary order
 TralingStopPrice		Returns a trailing stop price or null if none is in context of primary order
 Type		The type of order

[Top](#)

Methods

Name	Description
 Equals	Allow to compare two orders (Overrides Object.Equals(Object))



Finalize

(Overrides
[ObjectFinalize](#))



GetHashCode

(Overrides
[ObjectGetHashCode](#))



GetType

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

Operators

	Name	Description
	Equality(Order, Order)	Allow to compare two orders
	Inequality(Order, Order)	Allow to compare two orders

[Top](#)

See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

Order Properties

The [Order](#) type exposes the following members.

Properties

Name	Description
  Account	Account that has invoked an order.
  AverageFilledPrice	An average price of the order fill
  BoundTo	Obsolete. The if of bound order. It is related to OCO
  BoundToId	If it is a bound order.
  Comment	Returns notation for the selected order
  ExpirationTimeUtc	The date of expiration order in UTC for GTD orders

	FilledQuantity	Filled quantity
	ID	ID of an order
	Instrument	Represents the information of instrument for order
	IsActive	Checks if order is pending on server
	IsAutoTrade	Checks if order is invoked by strategy
	IsStopLossOrder	Returns true if specified order is a slave Stop Loss order in context of primary order
	IsTakeProfitOrder	Returns true if specified order is a slave Take Profit order in context of primary order
	LastUpdateTimeUtc	Returns last

		modification time for the order in UTC
 	Name	Custom name of an order given by user
 	PlacedTimeUtc	Date and time when the order was placed in UTC
 	Price	Price at which order can be executed
 	ProductType	Order Product Type
 	Quantity	Custom name of an order given by user
 	Side	Side of the order (Buy or Sell)
 	SLTPPriceType	Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute/tick or point
 	Status	Checks order

		status
	StopLossLimitPrice	Returns a stop-loss limit price or null if none is in context of primary order
	StopLossPrice	Returns a stop-loss price or null if none is in context of primary order
	StopPrice	Stop Price of the Limit order
	TakeProfitPrice	Returns a take-profit price based on SLTPPriceType or null if none is in context of primary order
	TIF	Order's Time-in-force
	TrailingStopOffset	Returns a trailing offset value or null if none is in context of primary order



TralingStopPrice

Returns a trailing stop price or null if none is in context of primary order



Type

The type of order

[Top](#)

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderAccount Property

Account that has invoked an order.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Account Account { get; } Copy
```

Property Value

[Account](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
AccountExample :
StrategyBuilder
{
    public
AccountExample()
    : base()
```

```
        {
            #region
Initialization

Credentials.ProjectName =
"AccountExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var count =
OrdersManager.GetOrders(false,
(order) => {
            return
order.Account.Name ==
AccountManager.Current.Name;
        }).Count;

if(OrdersManager.Count>0)

Notification.Print($"Current
Account has {count} pending
orders");
    }
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderAverageFilled Price Property

An average price of the order fill

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double
AverageFilledPrice { get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
AverageFilledPriceExample : StrategyBuilder
    {
        public
AverageFilledPriceExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "AverageFilledPriceExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;

orders.ForEach(order => {

Notification.Print($"Average
fill order price:
{order.AverageFilledPrice}");
})
}
}
}
```

See Also

Reference

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderBoundTo Property

Note: This API is now obsolete.

The if of bound order. It is related to OCO

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
[ObsoleteAttribute]  
public Order BoundTo { get; }
```

[Copy](#)

Property Value

[Order](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
        BoundToExample :  
            StrategyBuilder  
    {  
        public
```

[Copy](#)

```
    BoundToExample()
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"BoundToExample";
        #endregion
    }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    var orders =
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if (args
== TickStatus.IsBar)
{

var orderBound =
order.BoundTo;

if (orderBound!=null)

Notification.Print($"Order is
bound to {orderBound.ID}");
}
}
);
}
```

```
        }  
    }
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderBoundToId Property

If it is a bound order.

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public string BoundToId { get;  
}
```

[Copy](#)

Property Value
[String](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
using  
System.Collections.Generic;  
  
namespace TestEnv  
{  
    public class  
BoundToIdExample :  
StrategyBuilder  
{
```

[Copy](#)

```
    public
BoundToIdExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"BoundToIdExample";

Credentials.Description =
"BoundToId property Example";
    #endregion
}

    public override void
Init()
{
}

    public override void
Update(TickStatus args)
{
    var orders =
OrdersManager.GetOrders(true);
    foreach (Order
order in orders)
    {
        if
(string.IsNullOrEmpty(order.Bo
undToId))
            continue;

Notification.Print($"Order
{order.ID} is bound to order
{order.BoundToId}");
        var position =
PositionsManager.GetPositions(
```

```
    ) .Where(x => x.OpenOrderID ==  
order.BoundToId) .FirstOrDefault();  
  
        if (position  
!= null)  
  
Notification.Print($"Order  
{order.ID} is bound to  
position {position.ID}");  
    }  
}  
}  
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderComment Property

Returns notation for the selected order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Comment { get; } Copy
```

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
LastUpdateTimeUtcExample :
StrategyBuilder
{
    public
LastUpdateTimeUtcExample()
    : base()
```

```
        {
            #region
Initialization

Credentials.ProjectName =
"LastUpdateTimeUtcExample";
            #endregion
        }

    public override
void Init()
{
    var orderRequest
= new
OrderRequest(OrderType.Market,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1)
{
    Comment =
"Test"
};

OrdersManager.Send(orderReques
t);
}

    public override
void Update(TickStatus args)
{
    var order =
OrdersManager.GetOrders(SLTPIn
clude: false).Find(x =>
x.Comment == "Test");
    if(order!= null)
{

Notification.Comment($"
{order.ID}");
}
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderExpirationTime Utc Property

The date of expiration order in UTC for GTD orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime  
ExpirationTimeUtc { get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
ExpirationTimeUtcExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public  
ExpirationTimeUtcExample()  
        : base()  
{  
    #region  
Initialization  
  
Credentials.ProjectName =  
"ExpirationTimeUtcExample";  
    #endregion  
}  
  
        public override  
void Init()  
{  
  
}  
  
        public override  
void Update(TickStatus args)  
{  
    if (args ==  
TickStatus.IsBar)  
    {  
        var orders  
=  
OrdersManager.GetOrders(false)  
;  
  
orders.ForEach(order => {  
        if  
(order.ExpirationTimeUtc <  
DateTime.Now.AddHours(1))  
  
Notification.Print("Your  
pending order will be canceled  
within an hour");  
    } );  
}  
}
```

```
        }  
    }
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderFilledQuantity Property

Filled quantity

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double FilledQuantity {  
    get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
        FilledQuantityExample :  
            StrategyBuilder  
    {  
        public  
            FilledQuantityExample()
```

[Copy](#)

```
: base()
{
    #region
Initialization

Credentials.ProjectName =
"FilledQuantityExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(false);
;

orders.ForEach(order => {
    if
(order.FilledQuantity > 0)

Notification.Print($"Order
with id {order.ID} is
partially filled");
    } );
}
}

}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderID Property

ID of an order

Namespace: TradeApi.Trading

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string ID { get; }
```

[Copy](#)

Property Value

String

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class IDExample
        : StrategyBuilder
    {
        public IDExample()
            : base()
        {
            #region
            Initialization
        }
    }
}
```

[Copy](#)

```
Credentials.ProjectName =  
    "IDEExample";  
        #endregion  
    }  
  
    public override  
void Init()  
{  
  
}  
  
    public override  
void Update(TickStatus args)  
{  
    if (args ==  
TickStatus.IsBar)  
    {  
        var orders  
=  
OrdersManager.GetOrders(false)  
;  
  
orders.ForEach(order => {  
  
Notification.Print($"Available  
pending order id:  
{order.ID}");  
                } );  
    }  
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderInstrument Property

Represents the information of instrument for order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Instrument { Copy
    get; }
```

Property Value

[Instrument](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
InstrumentExample :
StrategyBuilder
{
    public
```

```
InstrumentExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"InstrumentExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if
(order.Instrument.Symbol ==
InstrumentsManager.Current.Sym
bol)

OrdersManager.Cancel(order);
} );
}
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderIsActive Property

Checks if order is pending on server

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool IsActive { get; } Copy
```

Property Value

Boolean

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
IsActiveExample :
StrategyBuilder
{
    public
IsActiveExample()
    : base()
```

```
        {
            #region
Initialization

Credentials.ProjectName =
"IsActiveExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if
(order.IsActive)

OrdersManager.Cancel(order);
});
}
}
}
```

See Also

Reference

Order Class

TradeApi.Trading Namespace

OrderIsAutoTrade Property

Checks if order is invoked by strategy

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool IsAutoTrade { get; } Copy
```

Property Value

[Boolean](#)

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
    IsAutoTradeExample : StrategyBuilder
    {
        public
        IsAutoTradeExample()
    }
}
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "IsAutoTradeExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if
(order.IsAutoTrade)

OrdersManager.Cancel(order);
}) ;
}
}
}
```

See Also

Reference

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderIsStopLossOrder Property

Returns true if specified order is a slave Stop Loss order in context of primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool IsStopLossOrder {  
    get; }
```

[Copy](#)

Property Value

Boolean

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
IsStopLossOrderExample :  
StrategyBuilder  
    {  
        public
```

[Copy](#)

```
    IsStopLossOrderExample()
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"IsStopLossOrderExample";
        #endregion
    }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if
(!order.IsStopLossOrder&&!orde
r.IsTakeProfitOrder)

OrdersManager.Cancel(order);
} );
}
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderIsTakeProfit Order Property

Returns true if specified order is a slave Take Profit order in context of primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool IsTakeProfitOrder
{ get; }
```

[Copy](#)

Property Value

Boolean

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
        IsTakeProfitOrderExample :
            StrategyBuilder
    {
        public
```

[Copy](#)

```
    IsTakeProfitOrderExample()
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "IsTakeProfitOrderExample";
        #endregion
    }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if
(!order.IsStopLossOrder&&!orde
r.IsTakeProfitOrder)

OrdersManager.Cancel(order);
});
}
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderLastUpdateTime Utc Property

Returns last modification time for the order in UTC

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime  
LastUpdateTimeUtc { get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
LastUpdateTimeUtcExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public
LastUpdateTimeUtcExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"LastUpdateTimeUtcExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;

orders.ForEach(order => {
    if
(order.LastUpdateTimeUtc.Hour
>
DateTime.Today.AddHours(6).Hour)

OrdersManager.Cancel(order);
    } );
}
}
```

```
        }  
    }
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderName Property

Custom name of an order given by user

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Name { get; }
```

[Copy](#)

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
NameExample : StrategyBuilder
    {
        public
NameExample()
            : base()
        {
            #region
Initialization
        }
    }
}
```

[Copy](#)

```
Credentials.ProjectName =
"NameExample";
#endregion
}

public override
void Init()
{
    var orderRequest
= new
OrderRequest(OrderType.Market,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1)
{
    Name =
"Rogue One"
};

OrdersManager.Send(orderRequest);
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
myOrder =
OrdersManager.GetOrders(false,
(order) => {
            return
order.Name == "Rogue One";
}).FirstOrDefault();
    }
}
```

```
        }  
    }
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderPlacedTimeUtc Property

Date and time when the order was placed in UTC

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime PlacedTimeUtc
{ get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System;
using System.Linq;
```

[Copy](#)

```
namespace TestEnv
{
    public class
PlacedTimeUtcExample :
StrategyBuilder
{
```

```
        public
PlacedTimeUtcExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"PlacedTimeUtcExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if
(order.PlacedTimeUtc.AddHours(
6) < DateTime.UtcNow)

OrdersManager.Cancel(order);
    });
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderPrice Property

Price at which order can be executed

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Price { get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
    PriceTypeExample : 
    StrategyBuilder
    {
        public
        PriceTypeExample()
            : base()
        {
            #region
```

[Copy](#)

```
Initialization

Credentials.ProjectName =
"PriceExample";
#endregion
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
instrument =
InstrumentsManager.Current;
        var
newRequest = new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Buy, quantity: 1);

newRequest.Price =
instrument.DayInfo.Ask;

OrdersManager.Send(newRequest)
;
    }
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderProductType Property

Order Product Type

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public ProductType ProductType
{ get; }
```

[Copy](#)

Property Value

[ProductType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
OrderProductTypeExample : StrategyBuilder
    {
        public
OrderProductTypeExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"OrderProductTypeExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if
(order.ProductType ==
ProductType.Intraday)

OrdersManager.Cancel(order);
});
}
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderQuantity Property

Custom name of an order given by user

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Quantity { get; }Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
NameExample : StrategyBuilder
    {
        public
NameExample()
            : base()
```

```
        {
            #region
Initialization

Credentials.ProjectName =
"NameExample";
            #endregion
        }

    public override
void Init()
{
    var orderRequest
= new
OrderRequest(OrderType.Market,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1)
{
    Name =
"Rogue One"
};

OrdersManager.Send(orderReques
t);
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var myOrder
=
OrdersManager.GetOrders(false,
(order) => {
    return
order.Quantity == 1;
}
}
```

```
    } ).FirstOrDefault();
}
}
}
```

▲ See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderSide Property

Side of the order (Buy or Sell)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public OrderSide Side { get; } Copy
```

Property Value

[OrderSide](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
SideExample : StrategyBuilder
    {
        public
SideExample()
            : base()
        {
            #region
Initialization
        }
    }
}
```

```
Credentials.ProjectName =
"SideExample";
#endregion
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if (order.Side ==
OrderSide.Buy)
    {

var updateRequest = new
ReplaceOrderRequest(order);

updateRequest.SLTPPriceType =
SLTPPriceType.Absolute;

updateRequest.StopLossPrice =
order.StopLossPrice ??
order.Price - 50 *
order.Instrument.MinimalTickSi
ze;

```

```
OrdersManager.Update(updateRequest);  
    }  
}  
});  
}  
}
```

See Also

Reference

Order Class

TradeApi.Trading Namespace

OrderSLTPPriceType Property

Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute/tick or point

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public SLTPPriceType  
SLTPPriceType { get; }
```

[Copy](#)

Property Value

[SLTPPriceType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;
```

[Copy](#)

```
namespace TestEnv  
{  
    public class  
SLTPPriceTypeExample :  
StrategyBuilder  
{
```

```
        public
SLTPPriceTypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"SLTPPriceTypeExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
instrument =
InstrumentsManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Buy, quantity: 1);

newRequest.SLTPPriceType =
SLTPPriceType.Absolute;

newRequest.TakeProfitPrice =
newRequest.Price + 50 *
instrument.MinimalTickSize;
}
```

```
OrdersManager.Send(newRequest)
;
}
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderStatus Property

Checks order status

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public OrderStatus Status {  
    get; }
```

[Copy](#)

Property Value

[OrderStatus](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
OrderStatusExample :  
StrategyBuilder  
    {  
        public  
OrderStatusExample()  
            : base()  
        {
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"OrderStatusExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if
(order.Status ==
OrderStatus.WaitingMarket)

OrdersManager.Cancel(order);
    });
}
}
}
```

See Also

Reference

Order Class

TradeApi.Trading Namespace

OrderStopLossLimit Price Property

Returns a stop-loss limit price or null if none is in context of primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double?  
StopLossLimitPrice { get; }
```

[Copy](#)

Property Value

[NullableDouble](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
StopLossLimitPriceExample :  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
StopLossLimitPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"StopLossLimitPriceExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    var
updateRequest = new
ReplaceOrderRequest(order);

updateRequest.SLTPPriceType =
SLTPPriceType.Absolute;

updateRequest.StopLossLimitPri
ce = order.StopLossLimitPrice
?? order.Price - 50 *
order.Instrument.MinimalTickSi
```

```
ze;

OrdersManager.Update(updateReq
uest);
} );
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderStopLossPrice Property

Returns a stop-loss price or null if none is in context of primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double? StopLossPrice { Copy
    get; }
```

Property Value

[NullableDouble](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
StopLossPriceExample :
StrategyBuilder
{
    public
```

```
StopLossPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"StopLossPriceExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    var
updateRequest = new
ReplaceOrderRequest(order);

updateRequest.SLTPPriceType =
SLTPPriceType.Absolute;

updateRequest.StopLossPrice =
order.StopLossPrice ??
order.Price - 50 *
order.Instrument.MinimalTickSi
```

```
ze;

OrdersManager.Update(updateReq
uest);
} );
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderStopPrice Property

Stop Price of the Limit order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double StopPrice { get;  
}
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
StopPriceExample :  
StrategyBuilder  
    {  
        public  
StopPriceExample()  
    }  
}
```

[Copy](#)

```
: base()
{
    #region
Initialization

Credentials.ProjectName =
"StopPriceExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    if (order.Status ==
OrderStatus.PendingNew)

Notification.Print($"Order is
stop price:
{order.StopPrice}");
}
)
}
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderTakeProfitPrice Property

Returns a take-profit price based on SLTPPriceType or null if none is in context of primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double? TakeProfitPrice
{ get; }
```

[Copy](#)

Property Value

[NullableDouble](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
TakeProfitPriceExample : StrategyBuilder
    {
```

[Copy](#)

```
        public
TakeProfitPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TakeProfitPriceExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
    var updateRequest = new
ReplaceOrderRequest(order);

updateRequest.TakeProfitPrice
= order.TakeProfitPrice ??
order.Price + 100 *
order.Instrument.MinimalTicksSi
ze;
}
}
```

```
    OrdersManager.Update(updateRequest);
    } );
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderTIF Property

Order's Time-in-force

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TimeInForce TIF { get; }
```

[Copy](#)

Property Value

[TimeInForce](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class TIFExample
        : StrategyBuilder
    {
        public TIFExample()
            : base()
        {
            #region
            Initialization
        }
    }
}
```

[Copy](#)

```
Credentials.ProjectName =
"TİFExample";
                                #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
instrument =
InstrumentsManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Market,
instrument,
AccountManager.Current,
OrderSide.Buy, quantity: 1);

newRequest.TIF =
TradeApi.Trading.TimeInForce.I
OC;

OrdersManager.Send(newRequest)
;
}
}
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderTrailingStopOffset Property

Returns a trailing offset value or null if none is in context of primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double?  
TrailingStopOffset { get; }
```

[Copy](#)

Property Value

[NullableDouble](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
TrailingStopOffsetExample :  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
    TrailingStopOffsetExample()
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "TrailingStopOffsetExample";
        #endregion
    }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
}

if (order.TrailingStopPrice > 0)
{
}

var updateRequest = new
ReplaceOrderRequest(order);

updateRequest.TrailingStopOffs
et = order.Price - 100 *
order.Instrument.MinimalTickSi
ze;
```

```
updateRequest.SLTPPriceType =  
    SLTPPriceType.Absolute;  
  
OrdersManager.Update(updateReq  
uest);  
    }  
}  
}  
}
```

See Also

Reference

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderTralingStopPrice Property

Returns a trailing stop price or null if none is in context of primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double?  
TralingStopPrice { get; }
```

[Copy](#)

Property Value

[NullableDouble](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;  
  
namespace TestEnv  
{  
    public class  
TralingStopPriceExample :  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
    TralingStopPriceExample()
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"TralingStopPriceExample";
        #endregion
    }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(true);

orders.ForEach(order =>
{
}

var updateRequest = new
ReplaceOrderRequest(order);

updateRequest.SLTPPriceType =
SLTPPriceType.Ticks;

updateRequest.TrailingStopOffs
et = 100;

OrdersManager.Update(updateReq
```

```
uest) ;  
        } ) ;  
    }  
}
```

See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderType Property

The type of order

Namespace: TradeApi.Trading

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public OrderType Type { get; } Copy
```

Property Value

OrderType

◀ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
TypeExample : StrategyBuilder
    {
        public
TypeExample()
            : base()
        {
            #region
Initialization
        }
    }
}
```

```
Credentials.ProjectName =
"TypeExample";
#endregion
}

public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
instrument =
InstrumentsManager.Current;
        var
newRequest = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current,
OrderSide.Buy, quantity: 1);

OrdersManager.Send(newRequest)
;
    }
}
}
```

See Also

Reference

[Order Class](#)

TradeApi.Trading Namespace

Order Methods

The [Order](#) type exposes the following members.

▲ Methods

Name	Description
 Equals	Allow to compare two orders (Overrides Object.Equals(Object))
 Finalize	(Overrides Object.Finalize)
 GetHashCode	(Overrides Object.GetHashCode)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

▲ See Also

Reference

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderEquals Method

Allow to compare two orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override bool Equals(Copy
                           Object obj
                           )
```

Parameters

obj [Object](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderFinalize Method

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
protected override void  
Finalize()
```

[Copy](#)

Implements
[ObjectFinalize](#)

► See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderGetHashCode Method

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public override int  
GetHashCode()
```

[Copy](#)

Return Value

[Int32](#)

◀ See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

Order Operators

The [Order](#) type exposes the following members.

► Operators

	Name	Description
 	Equality(Order, Order)	Allow to compare two orders
 	Inequality(Order, Order)	Allow to compare two orders

[Top](#)

► See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderEquality Operator

Allow to compare two orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator ==  
(  
    Order ord1,  
    Order ord2  
)
```

[Copy](#)

Parameters

ord1 [Order](#)

ord2 [Order](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderInequality Operator

Allow to compare two orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator !=  
(  
    Order ord1,  
    Order ord2  
)
```

[Copy](#)

Parameters

ord1 [Order](#)

ord2 [Order](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[Order Class](#)

[TradeApi.Trading Namespace](#)

OrderRequest Class

OrderRequest entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingOrderRequest](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
OrderRequest : IDisposable
```

[Copy](#)

The [OrderRequest](#) type exposes the following members.

▪ Constructors

	Name	Description
	OrderRequest	A shortcut to create basic order request with optional parameters.

[Top](#)

▪ Properties

Name	Description
 Account	Account that invoked an order (SINGLE + MULTIASSET)
 BoundTo	The if of bounded order. It is related to OCO
 Comment	Returns notation for the selected order
 ExpirationTimeUtc	The date of expiration order in UTC for GTD orders
 Instrument	An information of the instrument for order request
 Name	Custom name of an order given by user
 Price	Order's price

	ProductType	The ProductType of order
	Quantity	Order's quantity
	Side	Side of the order BUY or SELL
	SLTPPriceType	Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute or tick
	StopLossLimitPrice	Set price level of the stop loss limit order accompanied by primary order
	StopLossPrice	Set price level of the stop loss order accompanied by primary order
	StopPrice	Stop Price for

the StopLimit
order

	TakeProfitPrice	Set price level of take profit order accompanied by primary order
---	---------------------------------	---

	TIF	Order's time in force condition
---	---------------------	---------------------------------

	TrailingStopOffset	Set offset points to trigger Trailing Stop order accompanied by primary order
---	------------------------------------	---

	Type	The type of order
---	----------------------	-------------------

[Top](#)

Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)



Finalize

(Overrides
[ObjectFinalize](#))



[GetHashCode](#)

Serves as the default hash function.
(Inherited from [Object](#))



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

OrderRequest Constructor

A shortcut to create basic order request with optional parameters.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public OrderRequest(Copy
    OrderType orderType,
    Instrument instrument,
    Account account,
    OrderSide side,
    double quantity,
    double? price = null,
    double? stopPrice =
        null
)
```

Parameters

orderType [OrderType](#)

the order's type

instrument [Instrument](#)

the instrument on which order is based

account [Account](#)

the account on which order is based

side [OrderSide](#)

the order's side

quantity Double
 the order's quantity/lot
price NullableDouble (Optional)
 the order's target price (optional)
stopPrice NullableDouble (Optional)
 the order's stop price (optional)

Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
TypeExample: StrategyBuilder
    {
        public
TypeExample()
            : base()
        {
            #region
Initialization

Credentials.ProjectName =
"TypeExample";
            #endregion
        }

        public override
void Init()
        {

        }

        public override
void Update(TickStatus args)
        {
```

```
        if (args ==  
    TickStatus.IsBar)  
    {  
        var account =  
    AccountManager.Current;  
        var  
    newRequest = new  
    OrderRequest(orderType:OrderTy  
pe.Limit,  
  
    instrument:  
    InstrumentsManager.Current,  
  
    account:  
    AccountManager.Current,  
  
    side: OrderSide.Buy,  
  
    quantity: 1);  
  
    OrdersManager.Send(newRequest)  
;  
    }  
    }  
}
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequest Properties

The [OrderRequest](#) type exposes the following members.

Properties

Name	Description
  Account	Account that invoked an order (SINGLE + MULTIASSET)
  BoundTo	The if of bounded order. It is related to OCO
  Comment	Returns notation for the selected order
  ExpirationTimeUtc	The date of expiration order in UTC for GTD orders
  Instrument	An information

of the instrument for order request

	Name	Custom name of an order given by user
	Price	Order's price
	ProductType	The ProductType of order
	Quantity	Order's quantity
	Side	Side of the order BUY or SELL
	SLTPPriceType	Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute or tick
	StopLossLimitPrice	Set price level of the stop loss limit order accompanied

by primary
order

	StopLossPrice	Set price level of the stop loss order accompanied by primary order
	StopPrice	Stop Price for the StopLimit order
	TakeProfitPrice	Set price level of take profit order accompanied by primary order
	TIF	Order's time in force condition
	TrailingStopOffset	Set offset points to trigger Trailing Stop order accompanied by primary order
	Type	The type of order

[Top](#)

See Also

[Reference](#)

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestAccount Property

Account that invoked an order (SINGLE + MULTIASSET)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Account Account { get; set; }
```

[Copy](#)

Property Value

[Account](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
AccountExample :
StrategyBuilder
{
    public
AccountExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "AccountExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var account =
AccountManager.Current;
        var newRequest
= new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
account, OrderSide.Buy, 1);

OrdersManager.Send(newRequest)
;
    }
}
}
```

See Also

Reference

OrderRequest Class

TradeApi.Trading Namespace

OrderRequestBoundTo Property

The if of bounded order. It is related to OCO

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Order BoundTo { get;  
    set; }
```

[Copy](#)

Property Value
[Order](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
        BoundToExample :  
            StrategyBuilder  
    {  
        public  
        BoundToExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"BoundToExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders =
OrdersManager.GetOrders(false);
;
    if
(orders.Count > 0)

orders.ForEach(order =>
{
    var
account =
AccountManager.Current;
    var
instrument =
InstrumentsManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1);
}
```

```
newRequest.BoundTo = order;

OrdersManager.Send(newRequest)
;
}
}
}
}
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestCommentProperty

Returns notation for the selected order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Comment { get;  
    set; }
```

[Copy](#)

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
CommentExample :  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
CommentExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"CommentExample";
    #endregion
}

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var account =
AccountManager.Current;
    var newRequest
= new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
account, OrderSide.Buy, 1);

newRequest.Comment = $"An
order is placed based on
{this.Credentials.ProjectName}
strategy";

OrdersManager.Send(newRequest)
;
}
}
```

```
        }  
    }
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

Order RequestExpiration TimeUtc Property

The date of expiration order in UTC for GTD orders

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public DateTime  
ExpirationTimeUtc { get; set;  
}
```

[Copy](#)

Property Value
[DateTime](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
ExpirationTimeUtcExample :
```

[Copy](#)

```
StrategyBuilder
{
    public
ExpirationTimeUtcExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ExpirationTimeUtcExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var newRequest
= new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);

newRequest.ExpirationTimeUtc =
DateTime.Now.AddDays(1);
        newRequest.TIF
= TimeInForce.GTD;

OrdersManager.Send(newRequest)
;
}
```

```
        }  
    }  
}
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

Order RequestInstrument Property

An information of the instrument for order request

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Instrument { Copy
    get; set; }
```

Property Value
[Instrument](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
InstrumentExample : StrategyBuilder
{
```

```
        public
InstrumentExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"InstrumentExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var account =
AccountManager.Current;
        var instrument
= InstrumentsManager.Current;
        var newRequest
= new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1);

OrdersManager.Send(newRequest)
;
    }
}
}
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestName Property

Custom name of an order given by user

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Name { get; set; } Copy
```

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
NameExample : StrategyBuilder
    {
        public
NameExample()
            : base()
        {

```

```
        #region
Initialization

Credentials.ProjectName =
"NameExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var newRequest
= new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);

newRequest.Name = "My Order";

OrdersManager.Send(newRequest)
;
    }
}
}
```

See Also

Reference

OrderRequest Class

TradeApi.Trading Namespace

OrderRequestPrice Property

Order's price

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Price { get;  
    set; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
PriceExample : StrategyBuilder  
    {  
        public  
PriceExample()  
            : base()  
        {
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"PriceExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var account =
AccountManager.Current;
        var newRequest
= new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
account, OrderSide.Buy, 1);

newRequest.Price =
InstrumentsManager.Current.Day
Info.Ask + 50 *
InstrumentsManager.Current.Min
imalTickSize;

OrdersManager.Send(newRequest)
;
    }
}
}
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestProduct Type Property

The ProductType of order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public ProductType ProductType
{ get; set; }
```

[Copy](#)

Property Value

[ProductType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
ProductTypeExample:
StrategyBuilder
{
    public
ProductTypeExample()
    : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"ProductType Example";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
instrument =
InstrumentsManager.Current;
    var account =
AccountManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1);

newRequest.ProductType =
TradeApi.Trading.ProductType.I
ntraday;

OrdersManager.Send(newRequest)
;
}
}
```

```
        }  
    }
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestQuantity Property

Order's quantity

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double Quantity { get;  
    set; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
QuantityExample :  
StrategyBuilder  
    {  
        public  
QuantityExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"QuantityExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var quantity =
1;
    var newRequest
= new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, quantity);

OrdersManager.Send(newRequest)
;
}
}
}
```

See Also

Reference

OrderRequest Class

TradeApi.Trading Namespace

OrderRequestSide Property

Side of the order BUY or SELL

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public OrderSide Side { get;  
    set; }
```

[Copy](#)

Property Value

[OrderSide](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
        BoundToExample :  
            StrategyBuilder  
    {  
        public  
        BoundToExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"SideExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders =
OrdersManager.GetOrders(false);
;
    if
(orders.Count > 0)

orders.ForEach(order =>
{
    var
instrument =
InstrumentsManager.Current;
    var
side = OrderSide.Buy;
    var
newRequest = new
OrderRequest(OrderType.Limit,
instrument,
AccountManager.Current, side,
1);
}
```

```
OrdersManager.Send(newRequest)
;
}
}
}
}
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

Order RequestSLTPPriceType Property

Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute or tick

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public SLTPPriceType  
SLTPPriceType { get; set; }
```

[Copy](#)

Property Value

[SLTPPriceType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
SLTPPriceTypeExample:  
StrategyBuilder
```

[Copy](#)

```
        {
            public
SLTPPriceTypeExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"SLTPPriceTypeExample";
#endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
instrument =
InstrumentsManager.Current;
    var account =
AccountManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1);

newRequest.SLTPPriceType =
SLTPPriceType.Absolute;

newRequest.TakeProfitPrice =

```

```
InstrumentsManager.Current.Day  
Info.Ask + 50 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
newRequest.StopLossPrice =  
InstrumentsManager.Current.Day  
Info.Ask - 50 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
OrdersManager.Send(newRequest)  
;  
        }  
    }  
}  
}
```



See Also

[Reference](#)

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestStopLossLimitPrice Property

Set price level of the stop loss limit order accompanied by primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double  
StopLossLimitPrice { get; set;  
}
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
StopLossLimitPriceExample :  
StrategyBuilder
```

[Copy](#)

```
        {
            public
StopLossLimitPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"StopLossLimitPriceExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
instrument =
InstrumentsManager.Current;
    var account =
AccountManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1, 1);

newRequest.SLTPPriceType =
SLTPPriceType.Absolute;

newRequest.StopLossPrice =
```

```
InstrumentsManager.Current.Day  
Info.Ask - 50 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
newRequest.StopLossLimitPrice  
=  
InstrumentsManager.Current.Day  
Info.Ask - 100 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
OrdersManager.Send(newRequest)  
;  
}  
}  
}  
}
```



See Also

[Reference](#)

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestStopLossPrice Property

Set price level of the stop loss order accompanied by primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double StopLossPrice {  
    get; set; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
StopLossPriceExample :  
StrategyBuilder  
{  
    public  
StopLossPriceExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"StopLossPriceExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
instrument =
InstrumentsManager.Current;
    var account =
AccountManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1);

    newRequest.SLTPPriceType =
SLTPPriceType.Absolute;

    newRequest.StopLossPrice =
InstrumentsManager.Current.Day
Info.Ask - 50 *
InstrumentsManager.Current.Min
```

```
imAltTickSize;

OrdersManager.Send(newRequest)
;
}
}
}
}
```



▲ See Also

[Reference](#)

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestStop Price Property

Stop Price for the StopLimit order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double StopPrice { get;  
    set; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
StopPriceExample :  
StrategyBuilder  
{  
    public  
StopPriceExample()  
        : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"StopPriceExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var account =
AccountManager.Current;
    var newRequest
= new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
account, OrderSide.Buy, 1);

    newRequest.SLTPPriceType =
SLTPPriceType.Absolute;

    newRequest.StopPrice =
InstrumentsManager.Current.Day
Info.Ask - 100 *
InstrumentsManager.Current.Min
imalTickSize;

    OrdersManager.Send(newRequest)
;
```

```
        }  
    }  
}
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestTake ProfitPrice Property

Set price level of take profit order accompanied by primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public double TakeProfitPrice  
{ get; set; }
```

[Copy](#)

Property Value

[Double](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
        TakeProfitPriceExample:  
            StrategyBuilder  
    {  
        public  
        TakeProfitPriceExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
"TakeProfitPriceExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
instrument =
InstrumentsManager.Current;
        var account =
AccountManager.Current;
        var
newRequest = new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1);

        newRequest.SLTPPriceType =
SLTPPriceType.Absolute;

        newRequest.TakeProfitPrice =
InstrumentsManager.Current.Day
Info.Ask + 50 *
InstrumentsManager.Current.Min
```

```
imAltTickSize;

OrdersManager.Send(newRequest)
;
}
}
}
}
```



See Also

[Reference](#)

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestTIF Property

Order's time in force condition

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TimeInForce TIF { get;  
    set; }
```

[Copy](#)

Property Value

[TimeInForce](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
TimeInForceExample:  
StrategyBuilder  
    {  
        public  
TimeInForceExample()  
        : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"TimeInForceExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
instrument =
InstrumentsManager.Current;
    var account =
AccountManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1);

newRequest.TIF =
TimeInForce.Day;

OrdersManager.Send(newRequest)
;
}
}
```

```
        }  
    }
```

See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestTrailingStopOffset Property

Set offset points to trigger Trailing Stop order accompanied by primary order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double  
TrailingStopOffset { get; set;  
}
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
TrailingStopOffsetxample:  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
TrailingStopOffsetxample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TrailingStopOffsetxample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
instrument =
InstrumentsManager.Current;
    var account =
AccountManager.Current;
    var
newRequest = new
OrderRequest(OrderType.Limit,
instrument, account,
OrderSide.Buy, 1);

newRequest.SLTPPriceType =
SLTPPriceType.Absolute;

newRequest.TrailingStopOffset
=
InstrumentsManager.Current.Day
```

```
Info.Ask + 50 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
OrdersManager.Send(newRequest)  
;  
}  
}  
}  
}
```



See Also

Reference

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestType Property

The type of order

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public OrderType Type { get;  
    set; }
```

[Copy](#)

Property Value

[OrderType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
TypeExample: StrategyBuilder  
    {  
        public  
TypeExample()  
            : base()  
        {
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"TypeExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
instrument =
InstrumentsManager.Current;
        var account =
AccountManager.Current;
        var type =
OrderType.Limit;
        var
newRequest = new
OrderRequest(type, instrument,
account, OrderSide.Buy, 1);

OrdersManager.Send(newRequest)
;
    }
}
}
```

See Also

[Reference](#)

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequest Methods

The [OrderRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	Finalize	(Overrides ObjectFinalize)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a

string that
represents the
current object.
(Inherited from
[Object](#))

[Top](#)

◀ See Also

[Reference](#)

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderRequestFinalize Method

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
protected override void  
Finalize()
```

[Copy](#)

Implements
[ObjectFinalize](#)

◀ See Also

[Reference](#)

[OrderRequest Class](#)

[TradeApi.Trading Namespace](#)

OrderSide Enumeration

Order side type

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
[FlagsAttribute]  
public enum OrderSide
```

[Copy](#)

► Members

Member name	Value	Description
Buy	10,000	Buy
Sell	10,001	Sell

► See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

OrdersManager Class

OrdersManager entity

► Inheritance Hierarchy

[SystemObject](#) `TradeApi.TradingOrdersManager`

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class  
OrdersManager : [T:dNE=.QNQ=]
```

Copy

The `OrdersManager` type exposes the following members.

► Properties

	Name	Description
	<code>Count</code>	Returns number of all open orders excluding SL and TP
	<code>SLTPCount</code>	Returns number of all open Stop Loss and

Take Profit
orders



TotalBuyQuantity

Gets buy
quantity of all
pending
orders



TotalSellQuantity

Gets sell
quantity of all
pending
orders

[Top](#)

◀ Methods

	Name	Description
	Cancel(ListOrder)	Cancels orders with target and optional account parameter, resulting true if all are canceled successfully
	Cancel(Order)	Cancels an order synchronously
	CancelAsync	Cancels orders asynchronously with target and optional account parameter, resulting true if all are canceled successfully
	Equals	Determines whether the specified object is equal to the

		current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetOrders(Boolean)	Returns number of all open orders excluding SL and TP
	GetOrders(Boolean, PredicateOrder)	Gets a collection of all only opened orders. With optional delegate to grasp sl and tp orders, one side only and type filter. Access to the orders can be obtained by index.
	GetType	Gets the Type of the current instance. (Inherited from Object)
	Send	Sends trading order synchronously
	SendAsync	Sends trading order asynchronously with optional callback function
	ToString	Returns a string that represents the current object.

(Inherited from
[Object](#))

 	Update	Replace an order synchronously
 	UpdateAsync	Replaces an order asynchronously by modifying ReplaceOrderRequest with optional callback function

[Top](#)

Events

	Name	Description
 	OnCancel	The event at which the new order was canceled
 	OnFill	Occurs when the order was filled (FillSize args is partial or full)
 	OnPlace	The event at which the new order was placed
 	OnUpdate	The event at which the new order was updated

[Top](#)

See Also

[Reference](#)

TradeApi.Trading Namespace

OrdersManager Properties

The [OrdersManager](#) type exposes the following members.

Properties

	Name	Description
	Count	Returns number of all open orders excluding SL and TP
	SLTPCount	Returns number of all open Stop Loss and Take Profit orders
	TotalBuyQuantity	Gets buy quantity of all pending orders
	TotalSellQuantity	Gets sell quantity of all pending orders

[Top](#)

See Also

[Reference](#)

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerCount Property

Returns number of all open orders excluding SL and TP

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
CountExample : StrategyBuilder
    {
        public
CountExample()
            : base()
        {

```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"CountExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        if
(OrdersManager.Count > 0)
        {
            var
orders =
OrdersManager.GetOrders(true);
        }
    }
}
}
```

See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerSLTPCount Property

Returns number of all open Stop Loss and Take Profit orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double SLTPCount { get; } Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class SLTPCountExample :
        StrategyBuilder
    {
Copy
```

```
        public
SLTPCountExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"SLTPCountExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

if (OrdersManager.Count > 0)
    {

Notification.Print($"Total sl
and tp quantity
{OrdersManager.SLTPCount}");
    }
}
}
}
```

◀ See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerTotal BuyQuantity Property

Gets buy quantity of all pending orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double TotalBuyQuantity
{ get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
TotalBuyQuantityExample : 
StrategyBuilder
{
    public
TotalBuyQuantityExample()
    : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "TotalBuyQuantityExample";
            #endregion
        }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

        if (OrdersManager.Count>0)

Notification.Print($"Total buy
quantity
{OrdersManager.TotalBuyQuantit
y}");
    }
}
}
```

See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerTotal SellQuantity Property

Gets sell quantity of all pending orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double  
TotalSellQuantity { get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
TotalSellQuantityExample :  
StrategyBuilder  
{  
    public  
TotalSellQuantityExample()  
        : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "TotalSellQuantityExample";
            #endregion
        }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

        if (OrdersManager.Count > 0)

Notification.Print($"Total
sell quantity
{OrdersManager.TotalSellQuanti
ty}");
    }
}
}
```

See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManager Methods

The [OrdersManager](#) type exposes the following members.

▪ Methods

Name	Description
Cancel(ListOrder)	Cancels orders with target and optional account parameter, resulting true if all are canceled successfully
Cancel(Order)	Cancels an order synchronously
CancelAsync	Cancels orders asynchronously with target and optional account parameter, resulting true if all are canceled successfully
Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode	Serves as the default

		hash function. (Inherited from Object)
 	GetOrders(Boolean)	Returns number of all open orders excluding SL and TP
 	GetOrders(Boolean, PredicateOrder)	Gets a collection of all only opened orders. With optional delegate to grasp sl and tp orders, one side only and type filter. Access to the orders can be obtained by index.
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	Send	Sends trading order synchronously
 	SendAsync	Sends trading order asynchronously with optional callback function
 	ToString	Returns a string that represents the current object. (Inherited from Object)
 	Update	Replace an order synchronously

[UpdateAsync](#)

Replaces an order asynchronously by modifying ReplaceOrderRequest with optional callback function

[Top](#)

◀ See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerCancel Method

Overload List

Name	Description
 Cancel(ListOrder)	Cancels orders with target and optional account parameter, resulting true if all are canceled successfully
 Cancel(Order)	Cancels an order synchronously

[Top](#)

See Also

[Reference](#)

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

Orders

ManagerCancel(ListOrder) Method

Cancels orders with target and optional account parameter, resulting true if all are canceled successfully

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public bool Cancel(Copy
                  List<Order> orders
                )
```

Parameters

orders [ListOrder](#)

include sl and tp in count

Return Value

[Boolean](#)

▪ Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
```

```
namespace TestEnv
{
    public class
CancelExample :
StrategyBuilder
{
    public
CancelExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"CancelExample";
        #endregion
    }

    public override
void Init()
    {

    }

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

if (AccountManager.Current.Status == AccountStatus.Suspend)

OrdersManager.Cancel(OrdersManager.GetOrders(true));
    }
}
}
```

```
        }  
    }
```



▲ See Also

[Reference](#)

[OrdersManager Class](#)

[Cancel Overload](#)

[TradeApi.Trading Namespace](#)

Orders

ManagerCancel(Order)

Method

Cancels an order synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool Cancel(  
    Order order  
)
```

[Copy](#)

Parameters

order [Order](#)

Order to be canceled

Return Value

[Boolean](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System.Linq;
```

[Copy](#)

```
namespace TestEnv
{
    public class CancelExample : StrategyBuilder
    {
        public CancelExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "CancelExample";
            #endregion
        }

        public override void Init()
        {

        }

        public override void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar)
            {

                if (AccountManager.Current.Status ==
                    AccountStatus.Active)
                {
                    var
                    order =
                    OrdersManager.GetOrders(false,
                        x => x.Instrument ==
                            InstrumentsManager.Current).FirstOrDefault();
            }
        }
    }
}
```

```
        if  
        (order != null)  
        {  
  
            OrdersManager.Cancel(order:  
            order);  
        }  
    }  
}
```



See Also

Reference

[OrdersManager Class](#)

[Cancel Overload](#)

[TradeApi.Trading Namespace](#)

OrdersManagerCancel Async Method

Cancels orders asynchronously with target and optional account parameter, resulting true if all are canceled successfully

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public bool CancelAsync(Copy
    Order order,
    Action<bool> callback
= null
)
```

Parameters

order [Order](#)

Order to be canceled

callback [ActionBoolean](#) (Optional)

Method to be triggered as callback

Return Value

[Boolean](#)

◀ Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
CancelAsyncExample :
StrategyBuilder
{
    public
CancelAsyncExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"CancelAsyncExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

if (AccountManager.Current.Status == AccountStatus.Active)
    {
        var
order =
```

See Also

Reference

OrdersManager Class

TradeApi.Trading Namespace

OrdersManagerGet Orders Method

Overload List

Name	Description
 GetOrders(Boolean)	Returns number of all open orders excluding SL and TP
 GetOrders(Boolean, PredicateOrder)	Gets a collection of all only opened orders. With optional delegate to grasp sl and tp orders, one side only and type filter. Access to the orders can be obtained by index.

[Top](#)

See Also

[Reference](#)

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerGetOrders(Boolean) Method

Returns number of all open orders excluding SL and TP

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Order> GetOrders(Copy
    bool SLTPInInclude =
    true
)
```

Parameters

SLTPInInclude Boolean (Optional)
include sl and tp in count

Return Value

[ListOrder](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
```

```
namespace TestEnv
{
    public class GetOrdersExample : StrategyBuilder
    {
        public GetOrdersExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "GetOrdersExample";
            #endregion
        }

        public override void Init()
        {
        }

        public override void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar)
            {

                if (OrdersManager.Count > 0)
                {
                    var orders =
                        OrdersManager.GetOrders(SLTPin
                        clude: true);
                }
            }
        }
    }
}
```

```
        }  
    }  
}
```

See Also

[Reference](#)

[OrdersManager Class](#)

[GetOrders Overload](#)

[TradeApi.Trading Namespace](#)

OrdersManagerGetOrders(Boolean, PredicateOrder) Method

Gets a collection of all only opened orders. With optional delegate to grasp sl and tp orders, one side only and type filter. Access to the orders can be obtained by index.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Order> GetOrders(Copy
    bool SLTPinclude =
    true,
    Predicate<Order>
    predicate = null
)
```

Parameters

SLTPinclude Boolean (Optional)

include sl and tp in count

predicate PredicateOrder (Optional)

A delegate that contains a set of criteria and checks whether the passed parameter meets those criteria or not

Return Value

ListOrder

Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
GetOrdersExample : 
StrategyBuilder
{
    public
GetOrdersExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetOrdersExample";
        #endregion
    }

    public override
void Init()
    {

    }

    public override
void Update(TickStatus args)
    {
        if (args ==
TickStatus.IsBar)
        {
```

```
if(OrdersManager.Count>0)
{
    var
orders =
OrdersManager.GetOrders(SLTPin
clude: true, predicate: (x) =>
x.Quantity > 1);
}
}
```

See Also

[Reference](#)

[OrdersManager Class](#)

[GetOrders Overload](#)

[TradeApi.Trading Namespace](#)

OrdersManagerSend Method

Sends trading order synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TradeResult Send(Copy
    OrderRequest request
)
```

Parameters

request [OrderRequest](#)

Order's request main container

Return Value

[TradeResult](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;

namespace TestEnv
{
    public class
```

```
SendExample : StrategyBuilder
{
    public
SendExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SendExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

if (AccountManager.Current.Status == AccountStatus.Active)
    {
        var
request = new
OrderRequest(OrderType.Market,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);

OrdersManager.Send(request:
request);
    }
}
```

```
        }  
    }  
}
```



◀ See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerSend Async Method

Sends trading order asynchronously with optional callback function

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool SendAsync(Copy
    OrderRequest request,
    Action<TradeResult>
    callback = null
)
```

Parameters

request [OrderRequest](#)

Order's request main container

callback [ActionTradeResult](#) (Optional)

Method to be triggered as callback

Return Value

[Boolean](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
SendAsyncExample : StrategyBuilder
    {
        public
SendAsyncExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"SendAsyncExample";
            #endregion
        }

        public override void
Init()
        {

        }

        public override void
Update(TickStatus args)
        {
            if (args ==
TickStatus.IsBar)
            {

if (AccountManager.Current.Status
== AccountStatus.Active)
            {
                var
request = new
OrderRequest(OrderType.Market,
InstrumentsManager.Current,
```

```
AccountManager.Current,  
OrderSide.Buy, 1);  
  
OrdersManager.SendAsync(request:  
request, callback: (x) =>  
{  
    if  
(x.HasError)  
  
    Notification.Print($"Error:  
{x.Message}");  
});  
}  
}  
}  
}
```

See Also

[Reference](#)

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerUpdate Method

Replace an order synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public TradeResult Update(  
    ReplaceOrderRequest  
    request  
)
```

[Copy](#)

Parameters

request [ReplaceOrderRequest](#)
ReplaceOrderRequest to be
implemented to update an order

Return Value

[TradeResult](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv
```

[Copy](#)

```
    {
        public class
UpdateExample : 
StrategyBuilder
{
    public
UpdateExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"UpdateExample";
        #endregion
    }

    TradeResult result;

    public override
void Init()
{
}

private void
Show_Message(Order order)
{
    Notification.Print($"Order:
{order.ID} is updated");
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
```

```
=  
OrdersManager.GetOrders(false)  
;  
  
if(orders.Count > 0)  
  
    orders.ForEach(order =>  
        {  
  
            var updateRequest = new  
ReplaceOrderRequest(order);  
  
            updateRequest.Quantity = 2;  
  
            result =  
OrdersManager.Update(updateReq  
uest);  
            if  
(result.HasError)  
  
                Notification.Comment($"  
{result.Message}");  
            } );  
        }  
    }  
}
```

See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerUpdate Async Method

Replaces an order asynchronously by modifying ReplaceOrderRequest with optional callback function

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool UpdateAsync(Copy
    ReplaceOrderRequest
request,
    Action<TradeResult>
callback = null
)
```

Parameters

request [ReplaceOrderRequest](#)

Request to be implemented to update an order

callback [ActionTradeResult](#) (Optional)

Method to be triggered as callback

Return Value

[Boolean](#)

► Example

C#[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
UpdateAsyncExample :
StrategyBuilder
{
    public
UpdateAsyncExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"UpdateAsyncExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus
args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
orders =
OrdersManager.GetOrders (fa
lse);
```

```
if (orders.Count > 0)

    orders.ForEach(order =>
    {

        var updateRequest = new
ReplaceOrderRequest(order)
;

        updateRequest.Quantity =
2;

        OrdersManager.UpdateAsync(
updateRequest, (result)=>{

            if (result.HasError)

                Notification.Comment($""
{result.Message}");

            } );
        } );
    }
}
```

See Also

[Reference](#)

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManager Events

The [OrdersManager](#) type exposes the following members.

▲ Events

	Name	Description
 	OnCancel	The event at which the new order was canceled
 	OnFill	Occurs when the order was filled (FillSize args is partial or full)
 	OnPlace	The event at which the new order was placed
 	OnUpdate	The event at which the new order was updated

[Top](#)

▲ See Also

[Reference](#)

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerOn Cancel Event

The event at which the new order was canceled

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

▪ Syntax

C#

```
public event Action<Order>  
    OnCancel
```

[Copy](#)

Value
[ActionOrder](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
        OnCancelExample :  
            StrategyBuilder  
    {  
        public  
        OnCancelExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region Initialization
        Credentials.ProjectName =
"OnCancelExample";
        #endregion
    }

        public override
void Init()
{
    OrdersManager.OnCancel +=
Show_Message;
}

        private void
Show_Message(Order order)
{
    Notification.Print($"Order:
{order.ID} is canceled");
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(false);
;

    if(orders.Count > 0)

orders.ForEach(order =>
```

```
        {  
  
    OrdersManager.Cancel(order);  
    } );  
}  
  
    public override  
void Complete()  
{  
  
    OrdersManager.OnCancel -=  
Show_Message;  
    }  
}  
}
```

See Also

[Reference](#)

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerOnFill Event

Occurs when the order was filled (FillSize args is partial or full)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event Action<Order>  
OnFill
```

[Copy](#)

Value

[ActionOrder](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OnFillExample :  
StrategyBuilder  
{  
    public  
OnFillExample()
```

[Copy](#)

```
: base()
{
    #region
Initialization

Credentials.ProjectName =
"OnFillExample";
    #endregion
}

    public override
void Init()
{
    OrdersManager.OnFill +=
Show_Message;
}

    private void
Show_Message(Order order)
{
    Notification.Print($"Order:
{order.ID} is filled");
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
newRequest = new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);

OrdersManager.Send(newRequest)
```

```
;  
        }  
    }  
  
    public override  
void Complete()  
{  
  
    OrdersManager.OnFill -=  
    Show_Message;  
}  
}  
}
```

See Also

[Reference](#)

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerOnPlace Event

The event at which the new order was placed

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event Action<Order>
OnPlace
```

[Copy](#)

Value
[ActionOrder](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
OnPlaceExample :
StrategyBuilder
{
    public
OnPlaceExample()
    : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"OnPlaceExample";
            #endregion
        }

    public override
void Init()
{
    OrdersManager.OnPlace +=
Show_Message;
}

private void
Show_Message(Order order)
{
    Notification.Print($"Order:
{order.ID} is placed");
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
newRequest = new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);

OrdersManager.Send(newRequest)
;
```

```
        }

    public override
void Complete()
{
    OrdersManager.OnPlace -=
Show_Message;
}
}
```

See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrdersManagerOn Update Event

The event at which the new order was updated

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public event Action<Order>  
OnUpdate
```

[Copy](#)

Value
[ActionOrder](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OnUpdateExample :  
StrategyBuilder  
{  
    public  
OnUpdateExample()  
}
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "OnUpdateExample";
        #endregion
    }

        public override
void Init()
{
}

OrdersManager.OnUpdate +=  

Show_Message;
}

private void
Show_Message(Order order)
{

Notification.Print($"Order:
{order.ID} is updated");
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;

if(orders.Count > 0)

orders.ForEach(order =>
```

```
        {  
  
    var updateRequest = new  
ReplaceOrderRequest(order);  
  
    updateRequest.Quantity = 2;  
  
    OrdersManager.Update(updateReq  
uest);  
    } );  
}  
  
}  
  
public override  
void Complete()  
{  
  
    OrdersManager.OnUpdate -=  
Show_Message;  
    }  
}  
}
```

See Also

Reference

[OrdersManager Class](#)

[TradeApi.Trading Namespace](#)

OrderStatus Enumeration

Order's Status

Namespace: TradeApi.Trading

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public enum OrderStatus
```

[Copy](#)

► Members

Member name	Value	Description
PendingNew	1	PendingNew
PendingExecution	2	PendingExecution
PendingCancel	3	PendingCancel
PendingReplaceNotActive	5	PendingReplaceNotActive
New	10	New
Accepted	11	Accepted
PartiallyFilled	30	PartFilled
Filled	31	Filled
Canceled	40	Canceled
Refused	41	Refused
WaitingMarket	50	WaitingMarket
OffMarket	60	OffMarket
Unplaced	70	Unplaced

↳ See Also

Reference

[TradeApi.Trading Namespace](#)

OrdersViewer Class

OrdersViewer entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingOrdersViewer](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
OrdersViewer : [T:dNE=.QNQ=]
```

[Copy](#)

The [OrdersViewer](#) type exposes the following members.

▪ Properties

	Name	Description
	Count	Returns number of all open orders excluding SL and TP
	SLTPCount	Returns number of

all open
Stop Loss
and Take
Profit orders



TotalBuyQuantity

Gets buy
quantity of
all pending
orders



TotalSellQuantity

Gets sell
quantity of
all pending
orders

[Top](#)

◀ Methods

Name	Description
Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode	Serves as the default hash function. (Inherited from Object)
GetOrders(Boolean)	Returns

number of
all open
orders
excluding SL
and TP



[GetOrders\(Boolean, PredicateOrder\)](#)

Gets a collection of all only opened orders. With optional delegate to grasp sl and tp orders, one side only and type filter. Access to the orders can be obtained by index.



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ Events

	Name	Description
	OnCancel	The event at which the new order was canceled
	OnFill	Occurs when the order was filled (FillSize args is partial or full)
	OnPlace	The event at which the new order was placed
	OnUpdate	The event at which the new order was updated

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

OrdersViewer Properties

The [OrdersViewer](#) type exposes the following members.

Properties

	Name	Description
 	Count	Returns number of all open orders excluding SL and TP
 	SLTPCount	Returns number of all open Stop Loss and Take Profit orders
 	TotalBuyQuantity	Gets buy quantity of all pending orders
 	TotalSellQuantity	Gets sell quantity of all pending orders

[Top](#)

See Also

[Reference](#)

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerCount Property

Returns number of all open orders excluding SL and TP

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
CountExample :
IndicatorBuilder
{
    public
CountExample()
    : base()
```

[Copy](#)

```
{  
    #region  
    Initialization  
  
    Credentials.ProjectName =  
    "CountExample";  
    #endregion  
}  
  
    public override  
void Init()  
{  
  
}  
    public override  
void Update(TickStatus args)  
{  
    if (args ==  
    TickStatus.IsBar)  
    {  
        if  
(OrdersViewer.Count > 0)  
        {  
            var  
orders =  
OrdersViewer.GetOrders(true);  
        }  
    }  
}
```

See Also

Reference

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

Orders

ViewerSLTPCount

Property

Returns number of all open Stop Loss and Take Profit orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double SLTPCount { get; } Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class SLTPCountExample :
        IndicatorBuilder
    {
Copy
```

```
        public
SLTPCountExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"SLTPCountExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

if (OrdersViewer.Count > 0)
    {

Notification.Print($"Total sl
and tp quantity
{OrdersViewer.SLTPCount}");
    }
}
}
}
```

◀ See Also

Reference

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerTotalBuyQuantity Property

Gets buy quantity of all pending orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double TotalBuyQuantity
{ get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
TotalBuyQuantityExample : 
IndicatorBuilder
{
    public
TotalBuyQuantityExample()
    : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "TotalBuyQuantityExample";
            #endregion
        }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

        if (OrdersViewer.Count>0)

Notification.Print($"Total buy
quantity
{OrdersViewer.TotalBuyQuantity
}");
    }
}
}
```

See Also

Reference

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerTotalSellQuantity Property

Gets sell quantity of all pending orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double  
TotalSellQuantity { get; }
```

[Copy](#)

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
TotalSellQuantityExample :  
IndicatorBuilder  
    {  
        public  
TotalSellQuantityExample()  
        : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "TotalSellQuantityExample";
            #endregion
        }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

        if (OrdersViewer.Count > 0)

Notification.Print($"Total
sell quantity
{OrdersViewer.TotalSellQuantit
y}");
    }
}
}
```

See Also

Reference

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewer Methods

The [OrdersViewer](#) type exposes the following members.

▪ Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetOrders(Boolean)	Returns number of all open orders excluding SL and TP
 GetOrders(Boolean, PredicateOrder)	Gets a collection of all only

opened orders. With optional delegate to grasp sl and tp orders, one side only and type filter. Access to the orders can be obtained by index.



[GetType](#)

Gets the [Type](#) of the current instance.
(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerGet Orders Method

Overload List

Name	Description
 GetOrders(Boolean)	Returns number of all open orders excluding SL and TP
 GetOrders(Boolean, PredicateOrder)	Gets a collection of all only opened orders. With optional delegate to grasp sl and tp orders, one side only and type filter. Access to the orders can be obtained by index.

[Top](#)

See Also

[Reference](#)

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerGetOrders(Boolean) Method

Returns number of all open orders excluding SL and TP

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Order> GetOrders(  
    bool SLTPInInclude =  
    true  
)
```

[Copy](#)

Parameters

SLTPInInclude Boolean (Optional)
include sl and tp in count

Return Value

[ListOrder](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;
```

[Copy](#)

```
namespace TestEnv
{
    public class GetOrdersExample : IndicatorBuilder
    {
        public GetOrdersExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "GetOrdersExample";
            #endregion
        }

        public override void Init()
        {
        }

        public override void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar)
            {

                if (OrdersViewer.Count > 0)
                {
                    var orders =
                        OrdersViewer.GetOrders(SLTPinc
                            lude: true);
                }
            }
        }
    }
}
```

```
        }  
    }  
}
```

◀ See Also

[Reference](#)

[OrdersViewer Class](#)

[GetOrders Overload](#)

[TradeApi.Trading Namespace](#)

OrdersViewerGetOrders(Boolean, PredicateOrder) Method

Gets a collection of all only opened orders. With optional delegate to grasp sl and tp orders, one side only and type filter. Access to the orders can be obtained by index.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Order> GetOrders(Copy
    bool SLTPinclude =
    true,
    Predicate<Order>
    predicate = null
)
```

Parameters

SLTPinclude Boolean (Optional)

include sl and tp in count

predicate PredicateOrder (Optional)

A delegate that contains a set of criteria and checks whether the passed parameter meets those criteria or not

Return Value

ListOrder

Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
GetOrdersExample :
IndicatorBuilder
{
    public
GetOrdersExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetOrdersExample";
        #endregion
    }

    public override
void Init()
    {

    }

    public override
void Update(TickStatus args)
    {
        if (args ==
TickStatus.IsBar)
        {
```

Copy

```
if(OrdersViewer.Count>0)
{
    var
orders =
OrdersViewer.GetOrders(SLTPinc
lude: true, predicate: (x) =>
x.Quantity > 1);
}
}
```

See Also

[Reference](#)

[OrdersViewer Class](#)

[GetOrders Overload](#)

[TradeApi.Trading Namespace](#)

OrdersViewer Events

The [OrdersViewer](#) type exposes the following members.

► Events

	Name	Description
 	OnCancel	The event at which the new order was canceled
 	OnFill	Occurs when the order was filled (FillSize args is partial or full)
 	OnPlace	The event at which the new order was placed
 	OnUpdate	The event at which the new order was updated

[Top](#)

► See Also

[Reference](#)

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerOn Cancel Event

The event at which the new order was canceled

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public event Action<Order>  
    OnCancel
```

[Copy](#)

Value

[ActionOrder](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
        OnCancelExample :  
            IndicatorBuilder  
    {  
        public  
        OnCancelExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "OnCancelExample";
        #endregion
    }

        public override
void Init()
{
}

OrdersViewer.OnCancel +=  

Show_Message;
}

private void
Show_Message(Order order)
{

Notification.Print($"Order:
{order.ID} is canceled");
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersViewer.GetOrders(false);

if(orders.Count > 0)

orders.ForEach(order =>
{
}
```

```
        OrdersViewer.Cancel(order);
    });
}

public override
void Complete()
{
    OrdersViewer.OnCancel -=
    Show_Message;
}
}
```

See Also

[Reference](#)

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerOnFill Event

Occurs when the order was filled (FillSize args is partial or full)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event Action<Order>  
OnFill
```

[Copy](#)

Value

[ActionOrder](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OnFillExample :  
IndicatorBuilder  
    {  
        public  
OnFillExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "OnFillExample";
        #endregion
    }

        public override
void Init()
{
}

OrdersViewer.OnFill +=
Show_Message;
}

private void
Show_Message(Order order)
{

Notification.Print($"Order:
{order.ID} is filled");
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
newRequest = new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);

OrdersViewer.Send(newRequest);
```

```
        }

    public override
void Complete()
{
    OrdersViewer.OnFill -=
Show_Message;
}
}
```

See Also

Reference

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerOnPlace Event

The event at which the new order was placed

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event Action<Order>
OnPlace
```

[Copy](#)

Value
[ActionOrder](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
OnPlaceExample :
IndicatorBuilder
{
    public
OnPlaceExample()
    : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"OnPlaceExample";
            #endregion
        }

    public override
void Init()
{
    OrdersViewer.OnPlace +=
Show_Message;
}

private void
Show_Message(Order order)
{
    Notification.Print($"Order:
{order.ID} is placed");
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
newRequest = new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);

OrdersViewer.Send(newRequest);
    }
}
```

```
        }

    public override
void Complete()
{
    OrdersViewer.OnPlace -=
Show_Message;
}
}
```

See Also

[Reference](#)

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrdersViewerOnUpdate Event

The event at which the new order was updated

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public event Action<Order>  
    OnUpdate
```

[Copy](#)

Value

[ActionOrder](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
        OnUpdateExample :  
            IndicatorBuilder  
    {  
        public  
        OnUpdateExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "OnUpdateExample";
        #endregion
    }

        public override
void Init()
{
}

OrdersViewer.OnUpdate +=  

Show_Message;
}

private void
Show_Message(Order order)
{

Notification.Print($"Order:
{order.ID} is updated");
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersViewer.GetOrders(false);

if(orders.Count > 0)

orders.ForEach(order =>
{
}
```

```
    var updateRequest = new  
ReplaceOrderRequest(order);  
  
    updateRequest.Quantity = 2;  
  
    OrdersViewer.Update(updateRequ  
est);  
    } );  
}  
  
}  
  
public override  
void Complete()  
{  
  
    OrdersViewer.OnUpdate -=  
Show_Message;  
    }  
}  
}
```

See Also

[Reference](#)

[OrdersViewer Class](#)

[TradeApi.Trading Namespace](#)

OrderType Enumeration

Order's type

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
[FlagsAttribute]  
public enum OrderType
```

[Copy](#)

► Members

Member name	Value	Description
Limit	10,030	Market
Market	10,031	Market
StopLimit	10,033	StopLimit
TrailingStop	10,034	TrailingStop
Manual	10,070	Manual
Stop	10,073	Stop
OCO	10,076	OCO

See Also

Reference

[TradeApi.Trading Namespace](#)

Position Class

Position entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingPosition](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class Position :  
IDisposable
```

[Copy](#)

The [Position](#) type exposes the following members.

► Properties

	Name	Description
	Account	Provides account with which the position was opened
	GrossPnL	Profit/loss position ratio (without swaps or commissions)

based on the current broker's price. For open position it shows the profit/loss you would make if you close the position at the current price. If position closed, this parameter show profit/loss what trader have after closing this position.

	ID	Id of the position
	Instrument	Represents the information of instrument for position
	NetPnL	Profit/loss position ratio based on the current broker's price. For open position it shows the profit/loss you would make if you close the position at the current price. If

position closed,
this parameter
show
profit/loss what
trader have
after closing
this position.
During
backtesting this
value is equal
to GrossPnL
amid absence
of fee etc.

	OpenOrderID	Open order Id of the position
	OpenPrice	Price at which the position was opened
	OpenTimeUtc	Price at which the position was opened
	ProductType	Position Product Type
	Quantity	Position's quantity
	Side	Operation with which the position was opened(Long or Short)
	SLTPOrders	Collection of closing order which is bound

to primary
order



[SuperPositionId](#) SuperPosition
Id

[Top](#)

◀ Methods

	Name	Description
	ChangeProductType	Change Product Type for SubPosition
	Equals	Allow to compare two positions (Overrides Object.Equals(Object))
	Finalize	(Overrides Object.Finalize)
	GetHashCode	(Overrides Object.GetHashCode)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ Operators

	Name	Description
 	Equality(Position, Position)	Allow to compare two positions
 	Inequality(Position, Position)	Allow to compare two positions

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

Position Properties

The [Position](#) type exposes the following members.

Properties

	Name	Description
	Account	Provides account with which the position was opened
	GrossPnL	Profit/loss position ratio (without swaps or commissions) based on the current broker's price. For open position it shows the profit/loss you would make if you close the position at the current price. If position closed, this parameter

show
profit/loss
what trader
have after
closing this
position.

 	ID	Id of the position
 	Instrument	Represents the information of instrument for position
 	NetPnL	Profit/loss position ratio based on the current broker's price. For open position it shows the profit/loss you would make if you close the position at the current price. If position closed, this parameter show profit/loss what trader have after closing this

position.
During
backtesting
this value is
equal to
GrossPnL
amid
absence of
fee etc.

 	OpenOrderID	Open order Id of the position
 	OpenPrice	Price at which the position was opened
 	OpenTimeUtc	Price at which the position was opened
 	ProductType	Position Product Type
 	Quantity	Position's quantity
 	Side	Operation with which the position was opened(Long or Short)
 	SLTPOOrders	Collection of closing order

which is
bound to
primary
order



[SuperPositionId](#)

SuperPosition
Id

[Top](#)

▲ See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionAccount Property

Provides account with which the position was opened

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Account Account { get; } Copy
```

Property Value

[Account](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
AccountExample :
StrategyBuilder
{
    public
AccountExample()
Copy
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "AccountExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        double
allNetPnL = 0D;
        var
positions =
PositionsManager.GetPositions(
position =>
position.Account.Name ==
AccountManager.Current.Name);

positions.ForEach(position =>
allNetPnL += position.NetPnL);

Notification.Print($"Total PnL
by
{AccountManager.Current.Name}
:
{OrdersManager.TotalSellQuanti
```

```
    ty} " ) ;  
        }  
    }  
}
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionGrossPnL Property

Profit/loss position ratio (without swaps or commissions) based on the current broker's price. For open position it shows the profit/loss you would make if you close the position at the current price. If position closed, this parameter show profit/loss what trader have after closing this position.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double GrossPnL { get;  
}
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class
```

[Copy](#)

```
GrossPnLExample :  
StrategyBuilder  
{  
    public  
GrossPnLExample()  
        : base()  
{  
            #region  
Initialization  
  
Credentials.ProjectName =  
"GrossPnLExample";  
            #endregion  
}  
  
    public override  
void Init()  
{  
}  
  
    public override  
void Update(TickStatus args)  
{  
    if (args ==  
TickStatus.IsBar)  
    {  
        double  
totalGrossPnL = 0;  
        var  
positions =  
PositionsManager.GetPositions(  
);  
  
positions.ForEach(position =>  
{  
    totalGrossPnL +=  
position.GrossPnL;  
} );
```

```
Notification.Print($"Total  
GrossPnL {totalGrossPnL}");  
    }  
}  
}  
}
```



See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionID Property

Id of the position

Namespace: TradeApi.Trading

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string ID { get; }
```

[Copy](#)

Property Value

String

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System;
using
System.Collections.Generic;

namespace TestEnv
{
    public class IDEExample
: StrategyBuilder
    {
        public IDEExample()
: base()
{
    #region
```

[Copy](#)

```
Initialization

Credentials.ProjectName =
"IDExample";
                                #endregion
}

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
positions =
PositionsManager.GetPositions(
);
}

positions.ForEach(position =>
{
    Notification.Print($"Position
ID is {position.ID}");
} );
}
}
```

See Also

Reference

[Position Class](#)

TradeApi.Trading Namespace

PositionInstrument Property

Represents the information of instrument for position

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public Instrument Instrument { Copy
    get; }
```

Property Value

[Instrument](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
InstrumentExample :
StrategyBuilder
{
    public
InstrumentExample()
```

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "InstrumentExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
instrument =
InstrumentsManager.Current;
        var
positions =
PositionsManager.GetPositions(
position =>
position.Instrument.Symbol ==
instrument.Symbol);

positions.ForEach(position =>
{
    Notification.Print($"Position
Instrument ask value is
{position.Instrument.DayInfo.A
sk}");
})
;
```

```
        }  
    }  
}
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionNetPnL Property

Profit/loss position ratio based on the current broker's price. For open position it shows the profit/loss you would make if you close the position at the current price. If position closed, this parameter show profit/loss what trader have after closing this position. During backtesting this value is equal to GrossPnL amid absence of fee etc.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double NetPnL { get; }
```

[Copy](#)

Property Value

[Double](#)

► Example

C#

```
using System;
using Runtime.Script;
using TradeApi.Trading;
```

[Copy](#)

```
namespace TestEnv
```

```
{
```

```
        public class
NetPnLExample :
StrategyBuilder
{
    public
NetPnLExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"NetPnLExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
positions =
PositionsManager.GetPositions(
);
}

positions.ForEach(position =>
{
}

if(position.NetPnL > 0 &&
DateTime.UtcNow > new
DateTime(DateTime.UtcNow.Year,
DateTime.UtcNow.Month,
```

```
DateTime.UtcNow.Day, 16, 00,  
00))  
  
PositionsManager.Close(positio  
n);  
}  
}  
}  
}
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionOpenOrderID Property

Open order Id of the position

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string OpenOrderID {  
    get; }
```

[Copy](#)

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System;  
using  
System.Collections.Generic;  
  
namespace TestEnv  
{  
    public class  
OpenOrderIDExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public
OpenOrderIDExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"OpenOrderIDExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
);

positions.ForEach(position =>
{
    Notification.Print($"Position
OpenOrderID is
{position.OpenOrderID}");
}
);
    }
}
}
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionOpenPrice Property

Price at which the position was opened

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double OpenPrice { get; } Copy
```

Property Value

[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
OpenPriceExample : StrategyBuilder
    {
        public
OpenPriceExample()
            : base()
```

```
        {
            #region
Initialization

Credentials.ProjectName =
"OpenPriceExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
position => position.Side ==
PositionSide.Long);

positions.ForEach(position =>
{
    var stopPrice =
position.OpenPrice + 50 *
position.Instrument.MinimalTic
kSize;

OrdersManager.Send(new
OrderRequest(OrderType.Limit,
position.Instrument,
AccountManager.Current,
OrderSide.Sell,
```

```
position.Quantity,  
stopPrice, null));  
});  
}  
}  
}
```

See Also

Reference

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionOpenTimeUtc Property

Price at which the position was opened

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime OpenTimeUtc {  
    get; }
```

[Copy](#)

Property Value

[DateTime](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
        OpenTimeUtcExample :  
            StrategyBuilder  
    {  
        public  
            OpenTimeUtcExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "OpenTimeUtcExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
);
positions.ForEach(position =>
{
    if
(position.OpenTimeUtc.AddHours
(6) > DateTime.UtcNow)

PositionsManager.Close(positio
n);
} );
}
}
```

```
        }  
    }
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionProductType Property

Position Product Type

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public ProductType ProductType
{ get; }
```

[Copy](#)

Property Value
[ProductType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
ProductTypeExample : 
StrategyBuilder
{
    public
ProductTypeExample()
        : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
"ProductTypeExample";
            #endregion
        }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
position =>
position.ProductType ==
ProductType.Intraday);

Notification.Print($"Intraday
positions count:
{positions.Count}");
    }
}
}
```

See Also

Reference

Position Class

TradeApi.Trading Namespace

PositionQuantity Property

Position's quantity

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double Quantity { get; }Copy
```

Property Value
[Double](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
QuantityExample : StrategyBuilder
    {
        public
QuantityExample()
            : base()
```

```
        {
            #region
Initialization

Credentials.ProjectName =
"QuantityExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        double
quantity = 0D;
        var
positions =
PositionsManager.GetPositions(
);
}

positions.ForEach(position =>
quantity +=
position.Quantity);

Notification.Print($"Total
quantity by
{AccountManager.Current.Name}
: {quantity}");
    }
}
```

```
        }  
    }
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionSide Property

Operation with which the position was opened(Long or Short)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public PositionSide Side {  
    get; }
```

[Copy](#)

Property Value

[PositionSide](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
SideExample : StrategyBuilder  
    {  
        public  
SideExample()  
            : base()  
        {  
            #region
```

[Copy](#)

```
Initialization

Credentials.ProjectName =
"SideExample";
                                #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
position => position.Side ==
PositionSide.Long);

Notification.Print($"Position
buy side count:
{positions.Count}");
    }
}
}
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionSLTPOrders Property

Collection of closing order which is bound to primary order

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

▪ Syntax

C#

```
public List<Order> SLTPOrders
{ get; }
```

[Copy](#)

Property Value
[ListOrder](#)

▪ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using
System.Collections.Generic;

namespace TestEnv
{
    public class
SLTPOrdersExample :
StrategyBuilder
{
```

[Copy](#)

```
        public
SLTPOdersExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"SLTPOdersExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
);
        int count
= 0;

positions.ForEach(position =>
{
    position.SLTPOders.ForEach(or
der => count++);
} );
}

Notification.Print($"Total
count of sl / tp orders:
```

```
{ count }" );  
    }  
}  
}
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionSuperPosition Id Property

SuperPosition Id

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string SuperPositionId  
{ get; }
```

[Copy](#)

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
public class  
SuperPositionIdExample :  
StrategyBuilder  
{  
    private bool isOrderPlaced  
= false;  
  
    public  
SuperPositionIdExample() :  
base()
```

[Copy](#)

```
    {
        #region Initialization

        Credentials.ProjectName =
"SuperPositionIdExample";
        #endregion
    }

        public override void
Init()
{
}

PositionsManager.OnOpen +=
OnOpenPosition;
}

public override void
Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar &&
!isOrderPlaced)
    {
        var instrument =
InstrumentsManager.Current;
        var account =
AccountManager.Current;

            //Create market
order with Product Type -
Intraday
        OrderRequest
orderRequest = new
OrderRequest(OrderType.Market,
instrument, account,
OrderSide.Buy, 1);

orderRequest.ProductType =
ProductType.Intraday;
```

```
OrdersManager.SendAsync(orderRequest, PlaceOrderCallback);
    }
}

private void
PlaceOrderCallback(TradeResult
tradeResult)
{
    if
(!tradeResult.HasError)
        isOrderPlaced =
true;
}

private void
OnOpenPosition(Position
position)
{
    bool isSubPosition =
!string.IsNullOrEmpty(position
.SuperPositionId);
    if (isSubPosition)
    {
        //Change Product
        Type from Intraday to Delivery
        for SubPosition

        position.ChangeProductType(Pro
ductType.Delivery);
    }
    else
    {

        Notification.Print("It's
SuperPosition or simple
position");
    }
}
}
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

Position Methods

The [Position](#) type exposes the following members.

Methods

Name	Description
 ChangeProductType	Change Product Type for SubPosition
 Equals	Allow to compare two positions (Overrides Object.Equals(Object))
 Finalize	(Overrides Object.Finalize)
 GetHashCode	(Overrides Object.GetHashCode)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionChange ProductType Method

Change Product Type for SubPosition

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public void ChangeProductType(Copy
    ProductType
    productType
)
```

Parameters

productType [ProductType](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
public class
    ChangeProductTypeExample :
        StrategyBuilder
{
    private bool isOrderPlaced
    = false;
```

```
    public  
ChangeProductTypeExample() :  
base()  
{  
    #region Initialization  
  
    Credentials.ProjectName =  
    "ChangeProductTypeExample";  
    #endregion  
}  
  
    public override void  
Init()  
{  
  
    PositionsManager.OnOpen +=  
    OnOpenPosition;  
}  
  
    public override void  
Update(TickStatus args)  
{  
        if (args ==  
TickStatus.IsBar &&  
!isOrderPlaced)  
        {  
            var instrument =  
InstrumentsManager.Current;  
            var account =  
AccountManager.Current;  
  
                //Create market  
order with Product Type -  
Intraday  
            OrderRequest  
orderRequest = new  
OrderRequest(OrderType.Market,  
instrument, account,  
OrderSide.Buy, 1);
```

```
orderRequest.ProductType =
ProductType.Intraday;

OrdersManager.SendAsync(orderR
equest, PlaceOrderCallback);
    }
}

private void
PlaceOrderCallback(TradeResult
tradeResult)
{
    if
(!tradeResult.HasError)
        isOrderPlaced =
true;
}

private void
OnOpenPosition(Position
position)
{
    bool isSubPosition =
!string.IsNullOrEmpty(position
.SuperPositionId);
    if (isSubPosition)
    {
        //Change Product
        Type from Intraday to Delivery
        for SubPosition

        position.ChangeProductType(Pro
ductType.Delivery);
    }
    else
    {

Notification.Print("It's
SuperPosition or simple
```

```
    position");
}
}
```

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionEquals Method

Allow to compare two positions

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public override bool Equals(Copy
                           Object obj
                           )
```

Parameters

obj [Object](#)

Return Value

[Boolean](#)

► See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionFinalize Method

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
protected override void  
Finalize()
```

[Copy](#)

Implements
[ObjectFinalize](#)

► See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionGetHashCode Method

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public override int  
GetHashCode()
```

[Copy](#)

Return Value

[Int32](#)

◀ See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

Position Operators

The [Position](#) type exposes the following members.

Operators

	Name	Description
 	Equality(Position, Position)	Allow to compare two positions
 	Inequality(Position, Position)	Allow to compare two positions

[Top](#)

See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionEquality Operator

Allow to compare two positions

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public static bool operator ==  
(  
    Position pos1,  
    Position pos2  
)
```

[Copy](#)

Parameters

pos1 [Position](#)
pos2 [Position](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionInequality Operator

Allow to compare two positions

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public static bool operator !=  
(  
    Position pos1,  
    Position pos2  
)
```

[Copy](#)

Parameters

pos1 [Position](#)
pos2 [Position](#)

Return Value
[Boolean](#)

► See Also

[Reference](#)

[Position Class](#)

[TradeApi.Trading Namespace](#)

PositionSide Enumeration

Position side mode

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum PositionSide
```

[Copy](#)

► Members

Member name	Value	Description
Long	10,000	Long
Short	10,001	Short

► See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

PositionsManager Class

PositionsManager entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingPositionsManager](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

▪ Syntax

C#

```
public sealed class PositionsManager : IDisposable
```

[Copy](#)

The [PositionsManager](#) type exposes the following members.

▪ Properties

	Name	Description
 	Count	Returns number of all open positions

[Top](#)

▪ Methods

	Name	Description
 	Close	Close single position with optional quantity.
 	CloseAll	Close all

positions with optional account parameter. Returns an indication whether the positions are closed successfully



[CloseAsync](#)

Closes single position asynchronously with optional quantity and callback as result of asynchronous work



[Equals](#)

Determines whether the specified object is equal to the current object.
(Inherited from [Object](#))



[GetHashCode](#)

Serves as the default hash function.
(Inherited from [Object](#))



[GetPositions](#)

Returns a collection of all open positions. Access to the positions can be obtained by index.

 GetPositions(PredicatePosition)	Returns a collection of all open positions by given criteria. Access to the positions can be obtained by index.
 GetType	Gets the Type of the current instance. (Inherited from Object)
 RemoveStopLoss	Removes stop-loss order from given position synchronously
 RemoveStopLossAsync	Removes stop-loss order from given position asynchronously
 RemoveTakeProfit	Removes take profit order from given position synchronously
 RemoveTakeProfitAsync	Removes take profit order from given position asynchronously
 SetStopLoss	Sets a new (modifies existed) stop-loss order with

		given position synchronously
 	SetStopLossAsync	Sets a new (modifies existed) stop-loss order with given position asynchronously
 	SetTakeProfit	Sets a new (modifies existed) take-profit order with given position synchronously
 	SetTakeProfitAsync	Sets a new (modifies existed) take-profit order with given position synchronously
 	SetTrailingStop	Sets a new (modifies existed) trailing-top order with given position synchronously
 	SetTrailingStopAsync	Sets a new (modifies existed) trailing-top order with given position synchronously
 	ToString	Returns a

string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ Events

	Name	Description
 	OnClose	Occurs when the position was closed by user or by remote server
 	OnOpen	Occurs when the new position was placed

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

PositionsManager Properties

The [PositionsManager](#) type exposes the following members.

► Properties

	Name	Description
 	Count	Returns number of all open positions

[Top](#)

► See Also

[Reference](#)

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerCount Property

Returns number of all open positions

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
CountExample : StrategyBuilder
    {
        public
CountExample()
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"CountExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

if (PositionsManager.Count > 0)
{
    var
positions =
PositionsManager.GetPositions(
position => position.Side ==
PositionSide.Long);
}
    }
}
}
```

See Also

Reference

[PositionsManager Class](#)

TradeApi.Trading Namespace

PositionsManager Methods

The [PositionsManager](#) type exposes the following members.

▪ Methods

Name	Description
 Close	Close single position with optional quantity.
 CloseAll	Close all positions with optional account parameter. Returns an indication whether the positions are closed successfully
 CloseAsync	Closes single position asynchronously with optional quantity and callback as result of asynchronous work
 Equals	Determines whether the

specified object is equal to the current object.
(Inherited from [Object](#))

 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetPositions	Returns a collection of all open positions. Access to the positions can be obtained by index.
 	GetPositions(PredicatePosition)	Returns a collection of all open positions by given criteria. Access to the positions can be obtained by index.
 	GetType	Gets the Type of the current instance. (Inherited from Object)
 	RemoveStopLoss	Removes stop-loss order from given position synchronously
 	RemoveStopLossAsync	Removes stop-

		loss order from given position asynchronously
 	RemoveTakeProfit	Removes take profit order from given position synchronously
 	RemoveTakeProfitAsync	Removes take profit order from given position asynchronously
 	SetStopLoss	Sets a new (modifies existed) stop-loss order with given position synchronously
 	SetStopLossAsync	Sets a new (modifies existed) stop-loss order with given position asynchronously
 	SetTakeProfit	Sets a new (modifies existed) take-profit order with given position synchronously
 	SetTakeProfitAsync	Sets a new (modifies existed) take-profit order with given

		position synchronously
 	SetTrailingStop	Sets a new (modifies existed) trailing-top order with given position synchronously
 	SetTrailingStopAsync	Sets a new (modifies existed) trailing-top order with given position synchronously
 	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

Positions ManagerClose Method

Close single position with optional quantity.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool Close(Copy
    Position position,
    double? quantity =
    null
)
```

Parameters

position [Position](#)

queried position

quantity [NullableDouble](#) (Optional)

the lot size of the close order

Return Value

[Boolean](#)

► Example

C#

```
using System;
using Runtime.Script;
```

Copy

```
using TradeApi.Trading;

namespace TestEnv
{
    public class
CloseExample : StrategyBuilder
    {
        public
CloseExample()
            : base()
        {
            #region
Initialization

Credentials.ProjectName =
"CloseExample";
            #endregion
        }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
position =>
position.Instrument.Symbol ==
InstrumentsManager.Current.Sym
bol);

    if (positions.Count > 0)
```

```
positions.ForEach(position =>
{
    if
(position.OpenTimeUtc.AddHours
(6) > DateTime.UtcNow) {

    var result =
PositionsManager.Close(positio
n, position.Quantity);

    if(!result)

Notification.Print($"Bad try
to close position");
}

}
}
}
}
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

Positions ManagerCloseAll Method

Close all positions with optional account parameter. Returns an indication whether the positions are closed successfully

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool CloseAll(Copy
    Account account = null
)
```

Parameters

account [Account](#) (Optional)

An optional account parameter to specify the scope of position closing

Return Value

[Boolean](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
CloseAllExample : StrategyBuilder
    {
        public
CloseAllExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"CloseAllExample";
            #endregion
        }

        public override void
Init()
        {

        }

        public override void
Update(TickStatus args)
        {
            if (args ==
TickStatus.IsBar)
            {
                var positions
=
PositionsManager.GetPositions(posi
tion => position.Instrument.Symbol
==
InstrumentsManager.Current.Symbol)
;
        }
    }
}
```

```
    if(positions.Count>0)

        PositionsManager.CloseAll();
            }

        }

    }
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

Positions ManagerCloseAsync Method

Closes single position asynchronously with optional quantity and callback as result of asynchronous work

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public bool CloseAsync(Copy
    Position position,
    double? quantity =
    null,
    Action<bool> callback
    = null
)
```

Parameters

position [Position](#)

a position to be closed

quantity [NullableDouble](#) (Optional)

the lot size of the close order

callback [ActionBoolean](#) (Optional)

will be used after successful closing an operation

Return Value

Boolean

Example

C#

Copy

```
using System;
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
CloseAsyncExample : 
StrategyBuilder
{
    public
CloseAsyncExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"CloseAsyncExample";
        #endregion
    }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
```

```
        {  
            var  
positions =  
PositionsManager.GetPositions(  
position =>  
position.Instrument.Symbol ==  
InstrumentsManager.Current.Sym  
bol);  
  
if(positions.Count > 0)  
  
positions.ForEach(position =>  
{  
    if  
(position.OpenTimeUtc.AddHours  
(6) > DateTime.UtcNow)  
  
PositionsManager.CloseAsync(po  
sition, position.Quantity,  
(bool item) => {  
Notification.Print($"position  
is history");});  
});  
}  
}  
}  
}
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerGet Positions Method

▪ Overload List

Name	Description
 GetPositions	Returns a collection of all open positions. Access to the positions can be obtained by index.
 GetPositions(PredicatePosition)	Returns a collection of all open positions by given criteria. Access to the positions can be obtained by index.

[Top](#)

▪ See Also

[Reference](#)

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerGet Positions Method

Returns a collection of all open positions.
Access to the positions can be obtained by index.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Position>
GetPositions()
```

[Copy](#)

Return Value

[ListPosition](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
GetPositionsExample :
StrategyBuilder
{
    public
```

[Copy](#)

```
GetPositionsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetPositionsExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
positions =
PositionsManager.GetPositions(
);
}

if(positions.Count>0)

positions.ForEach(position =>
{

Notification.Print($"Position
ID is {position.ID}");
} );
}
}
}
```

See Also

Reference

[PositionsManager Class](#)

[GetPositions Overload](#)

[TradeApi.Trading Namespace](#)

PositionsManagerGet Positions(PredicatePosition) Method

Returns a collection of all open positions by given criteria. Access to the positions can be obtained by index.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public List<Position>
GetPositions(
    Predicate<Position>
predicate
)
```

[Copy](#)

Parameters

predicate [PredicatePosition](#)

A delegate that contains a set of criteria and checks whether the passed parameter meets those criteria or not

Return Value

[ListPosition](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
GetPositionsExample : 
StrategyBuilder
{
    public
GetPositionsExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetPositionsExample";
        #endregion
    }

    public override void
Init()
{
}

public override void
Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(po
sition =>
position.Instrument.Symbol ==
InstrumentsManager.Current.Symbo
l);
}
```

```
if (positions.Count>0)

    positions.ForEach(position => {

        Notification.Print($"Position ID
is {position.ID}");

    });
}
```

See Also

Reference

[PositionsManager Class](#)

[GetPositions Overload](#)

[TradeApi.Trading Namespace](#)

Positions ManagerRemoveStopLoss Method

Removes stop-loss order from given position synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool RemoveStopLoss(Copy
                           Position position
                         )
```

Parameters

position [Position](#)
queried position

Return Value
[Boolean](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
using System.Linq;
```

```
namespace TestEnv
{
    public class RemoveStopLossExample : StrategyBuilder
    {
        public RemoveStopLossExample() : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "RemoveStopLossExample";
            #endregion
        }

        public override void Init()
        {

        }

        public override void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar)
            {
                var
                positions =
                PositionsManager.GetPositions(
                );
            }

            if (positions.Count > 0)

                positions.ForEach(position =>
            {

```

```
        if  
        (position.SLTPOrders.Any(pos=>  
        pos.IsStopLossOrder))  
  
        PositionsManager.RemoveStopLos  
        s(position);  
    } );  
}  
}  
}
```

See Also

[Reference](#)

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

Positions ManagerRemoveStopLossAsync Method

Removes stop-loss order from given position asynchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool RemoveStopLossAsync(  
    Position position,  
    Action<bool> callback  
= null  
)
```

[Copy](#)

Parameters

position [Position](#)

queried position

callback [ActionBoolean](#) (Optional)

will be used after successful closing an operation

Return Value

[Boolean](#)

► Example

C#[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
RemoveStopLossAsyncExample : StrategyBuilder
    {
        public
RemoveStopLossAsyncExample()
            : base()
        {
            #region
Initialization

Credentials.ProjectName =
"RemoveStopLossAsyncExample";
            #endregion
        }

        public override
void Init()
        {

        }

        public override
void Update(TickStatus args)
        {
            if (args ==
TickStatus.IsBar)
            {
                var
positions =
PositionsManager.GetPositions(
);
            }
        }
    }
}
```

```
if(positions.Count>0)

positions.ForEach(position =>
{
    if
(position.SLTPOrders.Any(pos=>
pos.IsStopLossOrder))

PositionsManager.RemoveStopLos
sAsync(position, (bool item)
=> {

Notification.Print($"the stop
loss is history");

} );
}
}
}
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

Positions Manager RemoveTakeProfit Method

Removes take profit order from given position synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool RemoveTakeProfit(Copy
    Position position
)
```

Parameters

position [Position](#)
queried position

Return Value
[Boolean](#)

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;
using System.Linq;
```

```
namespace TestEnv
{
    public class RemoveTakeProfitExample : StrategyBuilder
    {
        public RemoveTakeProfitExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "RemoveTakeProfitExample";
            #endregion
        }

        public override void Init()
        {
        }

        public override void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar)
            {
                var
                positions =
                PositionsManager.GetPositions(
                );
            }

            if (positions.Count > 0)

                positions.ForEach(position =>
            {

```

```
        if  
        (position.SLTPOrders.Any(pos=>  
        pos.IsTakeProfitOrder) )  
  
        PositionsManager.RemoveTakePro  
        fit(position);  
    } );  
}  
}  
}
```

See Also

[Reference](#)

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

Positions Manager RemoveTake ProfitAsync Method

Removes take profit order from given position asynchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool RemoveTakeProfitAsync(  
    Position position,  
    Action<bool> callback  
= null  
)
```

[Copy](#)

Parameters

position [Position](#)

queried position

callback [ActionBoolean](#) (Optional)

will be used after successful operation

Return Value

[Boolean](#)

► Example

C#[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class RemoveTakeProfitAsyncExample : StrategyBuilder
    {
        public RemoveTakeProfitAsyncExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "RemoveTakeProfitAsyncExample";
            #endregion
        }

        public override void Init()
        {

        }

        public override void Update(TickStatus args)
        {
            if (args ==
                TickStatus.IsBar)
            {
                var
```

```
positions =
PositionsManager.GetPositi
ons();

if(positions.Count>0)

positions.ForEach(position
=> {
    if
(position.SLTPOrders.Any(p
os=>pos.IsTakeProfitOrder)
)

    PositionsManager.RemoveTak
eProfitAsync(position,
(bool item) => {
Notification.Print($"the
take profit is
history"); }) ;
} );
}
}
}
}
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerSet StopLoss Method

Sets a new (modifies existed) stop-loss order with given position synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool SetStopLoss(Copy
    Position position,
    double price
)
```

Parameters

position [Position](#)

a position to be set with pending stop loss

price [Double](#)

a price of the stop loss

Return Value

[Boolean](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
SetStopLossExample : 
StrategyBuilder
{
    public
SetStopLossExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SetStopLossExample";
        #endregion
    }

    public override void
Init()
    {

    }

    public override void
Update(TickStatus args)
    {
        if (args ==
TickStatus.IsBar)
        {
            var positions
= PositionsManager.GetPositions();

if (positions.Count > 0)

positions.ForEach(position => {
```

```
        if  
        (position.SLTPOrders.Any(pos=>!pos  
        .IsTakeProfitOrder))  
        {  
            var  
            price = position.OpenPrice +  
            InstrumentsManager.Current.Minimal  
            TickSize * 50;  
  
            PositionsManager.SetStopLoss (posit  
            ion, price);  
        }  
    }  
}
```

See Also

[Reference](#)

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerSet StopLossAsync Method

Sets a new (modifies existed) stop-loss order with given position asynchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool SetStopLossAsync(Copy
    Position position,
    double price,
    Action<bool> callback
    = null
)
```

Parameters

position [Position](#)

a position to be set with pending stop loss

price [Double](#)

a price of the stop loss

callback [ActionBoolean](#) (Optional)

will be used after successful operation

Return Value

[Boolean](#)

Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
SetStopLossAsyncExample :  
StrategyBuilder
{
    public
SetStopLossAsyncExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SetStopLossAsyncExample";
        #endregion
    }

    public override
void Init()
    {

    }

    public override
void Update(TickStatus args)
    {
        if (args ==
TickStatus.IsBar)
        {
            var
positions =
```

Copy

```
    PositionsManager.GetPositions()  
);  
  
    if(positions.Count>0)  
  
        positions.ForEach(position =>  
        {  
            if  
(position.SLTPOrders.Any(pos=>  
!pos.IsStopLossOrder))  
            {  
  
                var price = position.OpenPrice  
+  
InstrumentsManager.Current.Min  
imalTickSize * 50;  
  
                PositionsManager.SetStopLossAs  
ync(position, price, (bool  
item) => {  
Notification.Print($"the stop  
loss is set");});  
            }  
  
        } );  
    }  
}
```

See Also

[Reference](#)

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerSet TakeProfit Method

Sets a new (modifies existed) take-profit order with given position synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool SetTakeProfit(Copy
    Position position,
    double price
)
```

Parameters

position [Position](#)

a position to be set with pending take profit

price [Double](#)

a price of the take profit

Return Value

[Boolean](#)

► Example

C#

Copy

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
SetTakeProfitExample :
StrategyBuilder
{
    public
SetTakeProfitExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SetTakeProfitExample";
        #endregion
    }

    public override void
Init()
    {

    }

    public override void
Update(TickStatus args)
    {
        if (args ==
TickStatus.IsBar)
        {
            var positions
= PositionsManager.GetPositions();

if(positions.Count>0)

positions.ForEach(position => {
```

```
        if  
        (position.SLTPOrders.Any(pos=>!pos  
        .IsTakeProfitOrder))  
        {  
            var  
            price = position.OpenPrice -  
            InstrumentsManager.Current.Minimal  
            TickSize * 50;  
  
            PositionsManager.SetTakeProfit(pos  
            ition, price);  
        }  
  
    } ) ;  
}  
}  
}  
}
```



See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerSetTakeProfitAsync Method

Sets a new (modifies existed) take-profit order with given position synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool SetTakeProfitAsync(  
    Position position,  
    double price,  
    Action<bool> callback  
= null  
)
```

[Copy](#)

Parameters

position [Position](#)

a position to be set with pending take profit

price [Double](#)

a price of the take profit

callback [ActionBoolean](#) (Optional)

will be used after successful operation

Return Value

Boolean

Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class
SetTakeProfitAsyncExample : 
StrategyBuilder
{
    public
SetTakeProfitAsyncExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SetTakeProfitAsyncExample";
        #endregion
    }

    public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
```

Copy

```
        var  
positions =  
PositionsManager.GetPositions(  
);  
  
if(positions.Count > 0)  
  
    positions.ForEach(position =>  
{  
    if  
(position.SLTPOrders.Any(pos=>  
!pos.IsTakeProfitOrder))  
        {  
  
var price = position.OpenPrice  
-  
InstrumentsManager.Current.Min  
imalTickSize * 50;  
  
PositionsManager.SetTakeProfit  
Async(position, price, (bool  
item) => {  
Notification.Print($"the take  
profit is set");});  
        }  
  
    } );  
}  
}
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerSet TrailingStop Method

Sets a new (modifies existed) trailing-top order with given position synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool SetTrailingStop(Copy
    Position position,
    double price,
    int offset = 0
)
```

Parameters

position [Position](#)

a position to be set with trailing stop

price [Double](#)

a price of the trailing stop to start

offset [Int32](#) (Optional)

an offset to be trailed with active position

Return Value

[Boolean](#)

► Example

C#[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
SetTrailingStopExample :
StrategyBuilder
{
    public
SetTrailingStopExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"SetTrailingStopExample";
        #endregion
    }

    public override
void Init()
{
}

public override
void Update(TickStatus
args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositi
ons();
}
```

```
if (positions.Count > 0)

    positions.ForEach(position => {

        var price =
            position.OpenPrice -
            InstrumentsManager.Current
                .MinimalTickSize * 30;

        PositionsManager.SetTrailingStop(position, price,
            50);

    } );
}
}
```

See Also

Reference

PositionsManager Class

TradeApi.Trading Namespace

PositionsManagerSet TrailingStopAsync Method

Sets a new (modifies existed) trailing-top order with given position synchronously

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▲ Syntax

C#

```
public bool SetTrailingStopAsync(  
    Position position,  
    double price,  
    int offset = 0,  
    Action<bool> callback  
= null  
)
```

[Copy](#)

Parameters

position [Position](#)

a position to be set with trailing stop

price [Double](#)

a price of the trailing stop to start

offset [Int32](#) (Optional)

an offset to be trailed with active position

callback [ActionBoolean](#) (Optional)

will be used after successful operation

Return Value

Boolean

Example

C#

```
using Runtime.Script;
using TradeApi.Trading;
using System.Linq;

namespace TestEnv
{
    public class SetTrailingStopAsyncExample : StrategyBuilder
    {
        public SetTrailingStopAsyncExample()
            : base()
        {
            #region Initialization
            Credentials.ProjectName =
                "SetTrailingStopAsyncExample";
            #endregion
        }

        public override void Init()
        {

        }

        public override void Update(TickStatus args)
        {
```

Copy

```
        if (args ==  
    TickStatus.IsBar)  
        {  
            var  
positions =  
    PositionsManager.GetPositions()  
    ;  
  
        if(positions.Count>0)  
  
positions.ForEach(position =>  
{  
            var  
price = position.OpenPrice -  
InstrumentsManager.Current.Min  
imalTickSize * 50;  
  
    PositionsManager.SetTrailingSt  
opAsync(position, price, 50,  
(bool item) => {  
Notification.Print($"the  
trailing stop is set");});  
        } );  
    }  
}
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManager Events

The [PositionsManager](#) type exposes the following members.

▲ Events

	Name	Description
 	OnClose	Occurs when the position was closed by user or by remote server
 	OnOpen	Occurs when the new position was placed

[Top](#)

▲ See Also

[Reference](#)

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerOnClose Event

Occurs when the position was closed by user or by remote server

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public event Action<Position> OnClose
```

[Copy](#)

Value

[ActionPosition](#)

▪ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
    OnCloseExample :
    StrategyBuilder
    {
        public
        OnCloseExample()
    }
}
```

[Copy](#)

```
        : base()
    {
        #region Initialization

        Credentials.ProjectName =
"OnCloseExample";
        #endregion
    }

        public override
void Init()
{
    PositionsManager.OnClose +=
Show_Message;
}

        private void
Show_Message(Position
position)
{
    Notification.Print($"Position:
{position.ID} is closed");
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsManager.GetPositions(
position => position.Side ==
PositionSide.Long);

positions.ForEach(position =>
```

```
        {  
  
    PositionsManager.Close(position  
n);  
        } );  
    }  
  
    public override  
void Complete()  
{  
  
    PositionsManager.OnClose ==  
Show_Message;  
    }  
}  
}
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsManagerOn Open Event

Occurs when the new position was placed

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event Action<Position>  
OnOpen
```

[Copy](#)

Value

[ActionPosition](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OnOpenExample :  
StrategyBuilder  
{  
    public  
OnOpenExample()  
        : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"OnOpenExample";
            #endregion
        }

    public override
void Init()
{
    PositionsManager.OnOpen +=
Show_Message;
}

private void
Show_Message(Position
position)
{
    Notification.Print($"Position:
{position.ID} is opened");
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
newRequest = new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);

OrdersManager.Send(newRequest)
```

```
;  
    }  
  
    public override  
void Complete()  
{  
  
    PositionsManager.OnOpen -=  
    Show_Message;  
}  
}  
}
```

See Also

Reference

[PositionsManager Class](#)

[TradeApi.Trading Namespace](#)

PositionsViewer Class

PositionsViewer entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingPositionsViewer](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class  
    PositionsViewer : IDisposable
```

Copy

The [PositionsViewer](#) type exposes the following members.

► Properties

	Name	Description
	Count	Returns number of all open positions

[Top](#)

► Methods

	Name	Description
	Equals	Determines whether the specified

object is equal to the current object.
(Inherited from [Object](#))

 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetPositions	Returns a collection of all open positions. Access to the positions can be obtained by index.
 	GetPositions(PredicatePosition)	Returns a collection of all open positions by given criteria. Access to the positions can be obtained by index.
	GetType	Gets the Type of the current instance.

(Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

Events

Name	Description
OnClose	Occurs when the position was closed by user or by remote server
OnOpen	Occurs when the new position was placed

[Top](#)

See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

PositionsViewer Properties

The [PositionsViewer](#) type exposes the following members.

Properties

	Name	Description
 	Count	Returns number of all open positions

[Top](#)

See Also

[Reference](#)

[PositionsViewer Class](#)

[TradeApi.Trading Namespace](#)

PositionsViewerCount Property

Returns number of all open positions

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public int Count { get; }
```

[Copy](#)

Property Value

[Int32](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
CountExample :
IndicatorBuilder
{
    public
CountExample()
    : base()
    {
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"CountExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

if (PositionsViewer.Count > 0)
{
    var
positions =
PositionsViewer.GetPositions(p
osition => position.Side ==
PositionSide.Long);
}
}
}
}
```

See Also

[Reference](#)

[PositionsViewer Class](#)

[TradeApi.Trading Namespace](#)

PositionsViewer Methods

The [PositionsViewer](#) type exposes the following members.

Methods

Name	Description
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
  GetPositions	Returns a collection of all open positions. Access to the positions can be obtained by index.
  GetPositions(PredicatePosition)	Returns a collection of

all open positions by given criteria. Access to the positions can be obtained by index.



[GetType](#)

Gets the [Type](#) of the current instance. (Inherited from [Object](#))



[ToString](#)

Returns a string that represents the current object. (Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[PositionsViewer Class](#)

[TradeApi.Trading Namespace](#)

PositionsViewerGet Positions Method

▪ Overload List

Name	Description
 GetPositions	Returns a collection of all open positions. Access to the positions can be obtained by index.
 GetPositions(PredicatePosition)	Returns a collection of all open positions by given criteria. Access to the positions can be obtained by index.

[Top](#)

▪ See Also

[Reference](#)

[PositionsViewer Class](#)

[TradeApi.Trading Namespace](#)

PositionsViewerGet Positions Method

Returns a collection of all open positions.
Access to the positions can be obtained by index.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public List<Position>
GetPositions()
```

[Copy](#)

Return Value

[ListPosition](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
GetPositionsExample :
IndicatorBuilder
{
    public
```

[Copy](#)

```
GetPositionsExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"GetPositionsExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var
positions =
PositionsViewer.GetPositions()
;

if(positions.Count>0)

positions.ForEach(position =>
{

Notification.Print($"Position
ID is {position.ID}");
} );
}
}
}
```

See Also

Reference

[PositionsViewer Class](#)

[GetPositions Overload](#)

[TradeApi.Trading Namespace](#)

PositionsViewerGet Positions(PredicatePosition) Method

Returns a collection of all open positions by given criteria. Access to the positions can be obtained by index.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public List<Position>
GetPositions(
    Predicate<Position>
predicate
)
```

[Copy](#)

Parameters

predicate [PredicatePosition](#)

A delegate that contains a set of criteria and checks whether the passed parameter meets those criteria or not

Return Value
[ListPosition](#)

► Example

C#

[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
GetPositionsExample : 
IndicatorBuilder
{
    public
GetPositionsExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"GetPositionsExample";
        #endregion
    }

    public override void
Init()
{
}

public override void
Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsViewer.GetPositions(pos
ition =>
position.Instrument.Symbol ==
InstrumentsManager.Current.Symbo
l);
}
```

```
if(positions.Count>0)

positions.ForEach(position => {

    Notification.Print($"Position ID
is {position.ID}");

    }
}

}
```

See Also

Reference

[PositionsViewer Class](#)

[GetPositions Overload](#)

[TradeApi.Trading Namespace](#)

PositionsViewer Events

The [PositionsViewer](#) type exposes the following members.

▲ Events

	Name	Description
 	OnClose	Occurs when the position was closed by user or by remote server
 	OnOpen	Occurs when the new position was placed

[Top](#)

▲ See Also

[Reference](#)

[PositionsViewer Class](#)

[TradeApi.Trading Namespace](#)

PositionsViewerOnClose Event

Occurs when the position was closed by user or by remote server

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event Action<Position>  
OnClose
```

[Copy](#)

Value

[ActionPosition](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OnCloseExample :  
IndicatorBuilder  
    {  
        public  
OnCloseExample()  
    }  
}
```

[Copy](#)

```
        : base()
    {
        #region Initialization

        Credentials.ProjectName =
"OnCloseExample";
        #endregion
    }

        public override
void Init()
{
    PositionsViewer.OnClose +=

Show_Message;
}

        private void
Show_Message(Position
position)
{
    Notification.Print($"Position:
{position.ID} is closed");
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
positions =
PositionsViewer.GetPositions(p
osition => position.Side ==
PositionSide.Long);

positions.ForEach(position =>
```

```
        {  
  
    PositionsViewer.Close(position  
    ) ;  
    } ) ;  
}  
}  
  
public override  
void Complete()  
{  
  
    PositionsViewer.OnClose -=  
    Show_Message;  
    }  
}  
}
```

See Also

Reference

[PositionsViewer Class](#)

[TradeApi.Trading Namespace](#)

PositionsViewerOn Open Event

Occurs when the new position was placed

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public event Action<Position>  
OnOpen
```

[Copy](#)

Value

[ActionPosition](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OnOpenExample :  
IndicatorBuilder  
    {  
        public  
OnOpenExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"OnOpenExample";
            #endregion
        }

    public override
void Init()
{
}

PositionsViewer.OnOpen +=
Show_Message;
}

private void
Show_Message(Position
position)
{
}

Notification.Print($"Position:
{position.ID} is opened");
}

public override
void Update(TickStatus args)
{
}

public override
void Complete()
{

}

PositionsViewer.OnOpen -=
Show_Message;
}
}
```

See Also

Reference

[PositionsViewer Class](#)

[TradeApi.Trading Namespace](#)

ProductType Enumeration

Product Type

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum ProductType
```

[Copy](#)

► Members

Member name	Value	Description
General	0	General
Intraday	1	Intraday
Delivery	2	Delivery

► See Also

Reference

[TradeApi.Trading Namespace](#)

ReplaceOrderRequest Class

ModifyOrderRequest entity

► Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingReplaceOrderRequest](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public sealed class  
ReplaceOrderRequest :  
IDisposable
```

[Copy](#)

The [ReplaceOrderRequest](#) type exposes the following members.

► Constructors

Name	Description
 ReplaceOrderRequest(Order)	Creates replace order request
 ReplaceOrderRequest(Order, Double)	Creates request for

changing price.

	ReplaceOrderRequest(Order, Double, Double)	Creates request for changing price and amount.
---	--	--

[Top](#)

► Properties

	Name	Description
	BoundTo	The order which current order is bound to; if the former is executed, the system cancels the latter. Used by OCO orders.
	Comment	Comment for the order.
	ExpirationTime	The date of expiration order (for GTD orders).
	Order	Modifying order
	Price	Order's price

	Quantity	Value of the order in lots.
	SLTPPriceType	Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute or tick
	StopLossLimitPrice	Set price level of the stop loss limit order accompanied by primary order
	StopLossPrice	Set price level of the stop loss order accompanied by primary order
	StopPrice	Stop Price for the StopLimit order
	TakeProfitPrice	Set price level of take profit order accompanied

by primary
order



TIF

Order's
Time-in-
force.



TrailingStopOffset

Offset points
to trigger
the trailing
stop loss.

[Top](#)

Methods

	Name	Description
≡	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
≡	Finalize	(Overrides ObjectFinalize)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetType	Gets the Type of the current instance. (Inherited from Object)



ToString

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

ReplaceOrderRequest Constructor

↳ Overload List

Name	Description
 ReplaceOrderRequest(Order)	Creates replace order request
 ReplaceOrderRequest(Order, Double)	Creates request for changing price.
 ReplaceOrderRequest(Order, Double, Double)	Creates request for changing price and amount.

[Top](#)

↳ See Also

[Reference](#)

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

ReplaceOrderRequest(Order) Constructor

Creates replace order request

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public ReplaceOrderRequest (Copy
    Order order
)
```

Parameters

order [Order](#)
Order to be replaced

► Example

C#

```
using Runtime.Script; Copy
using TradeApi.Trading;

namespace TestEnv
{
    public class
ReplaceOrderRequestExample :
```

```
StrategyBuilder
{
    public
ReplaceOrderRequestExample()
    : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ReplaceOrderRequestExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;

if(orders.Count > 0)

orders.ForEach(order =>
{
    var updateRequest = new
ReplaceOrderRequest(order);

updateRequest.Quantity = 2;
}
```

```
OrdersManager.UpdateAsync(updateRequest, (result)=>{

    if (result.HasError)

        Notification.Comment($""
            {result.Message}");

    } );
}

}

}

}
```

See Also

[Reference](#)

[ReplaceOrderRequest Class](#)

[ReplaceOrderRequest Overload](#)

[TradeApi.Trading Namespace](#)

ReplaceOrderRequest(Order, Double) Constructor

Creates request for changing price.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public ReplaceOrderRequest (Copy
    Order order,
    double price
)
```

Parameters

order [Order](#)

Order to be replaced

price [Double](#)

new order price

◀ Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
```

```
    {
        public class ReplaceOrderRequestExample : StrategyBuilder
        {
            public ReplaceOrderRequestExample() : base()
            {
                #region Initialization
                Credentials.ProjectName =
                    "ReplaceOrderRequestExample";
                #endregion
            }

            public override void Init()
            {

            }

            public override void Update(TickStatus args)
            {
                if (args ==
                    TickStatus.IsBar)
                {
                    var orders =
                        OrdersManager.GetOrders(false);
                }

                if (orders.Count > 0)
                    orders.ForEach(order =>
                {
                    var price =
```

```
order.SLTPPriceType ==  
SLTPPriceType.Absolute ?  
  
    order.Price + 50 *  
    InstrumentsManager.Current.Min  
    imalTickSize : 50;  
  
    var updateRequest = new  
    ReplaceOrderRequest(order,  
    price);  
  
    updateRequest.Quantity = 2;  
  
    OrdersManager.UpdateAsync(updateRequest, (result) =>{  
  
        if (result.HasError)  
  
            Notification.Comment($"  
            {result.Message}");  
  
    } );  
};  
}  
}  
}
```

See Also

Reference

[ReplaceOrderRequest Class](#)

[ReplaceOrderRequest Overload](#)

[TradeApi.Trading Namespace](#)

ReplaceOrderRequest(Order, Double, Double) Constructor

Creates request for changing price and amount.

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

◀ Syntax

C#

```
public ReplaceOrderRequest (Copy
    Order order,
    double price,
    double amount
)
```

Parameters

order [Order](#)
modifying order

price [Double](#)
new order price
amount [Double](#)
new order amount

◀ Example

C#[Copy](#)

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
ReplaceOrderRequestExample : 
StrategyBuilder
{
    public
ReplaceOrderRequestExample()
        : base()
    {
        #region
Initialization

Credentials.ProjectName =
"ReplaceOrderRequestExample";
        #endregion
    }

    public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;
}
```

```
if(orders.Count > 0)

    orders.ForEach(order =>
        {

            var price =
                order.SLTPPriceType ==
                SLTPPriceType.Absolute ? 

                    order.Price + 50 *
                    InstrumentsManager.Current.Min
                    imalTickSize : 50;

            var updateRequest = new
                ReplaceOrderRequest(order,
                    price, 2);

            OrdersManager.UpdateAsync(updateRequest, (result)=>{

                if (result.HasError)

                    Notification.Comment(${result.Message});

                } );
            });

        }
    });
}
```

See Also

Reference

[ReplaceOrderRequest Class](#)

[ReplaceOrderRequest Overload](#)

[TradeApi.Trading Namespace](#)

ReplaceOrderRequest Properties

The [ReplaceOrderRequest](#) type exposes the following members.

Properties

	Name	Description
	BoundTo	The order which current order is bound to; if the former is executed, the system cancels the latter. Used by OCO orders.
	Comment	Comment for the order.
	ExpirationTime	The date of expiration order (for GTD orders).
	Order	Modifying order

	Price	Order's price
	Quantity	Value of the order in lots.
	SLTPPriceType	Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute or tick
	StopLossLimitPrice	Set price level of the stop loss limit order accompanied by primary order
	StopLossPrice	Set price level of the stop loss order accompanied by primary order
	StopPrice	Stop Price for the StopLimit order
	TakeProfitPrice	Set price level of take

profit order
accompanied
by primary
order



[TIF](#)

Order's
Time-in-
force.



[TrailingStopOffset](#)

Offset points
to trigger
the trailing
stop loss.

[Top](#)

▲ See Also

[Reference](#)

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

ReplaceOrder RequestBoundTo Property

The order which current order is bound to; if the former is executed, the system cancels the latter. Used by OCO orders.

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

▪ Syntax

C#

```
public Order BoundTo { get;  
    set; }
```

[Copy](#)

Property Value
[Order](#)

▪ Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
BoundToExample :  
StrategyBuilder
```

[Copy](#)

```
        {
            public
        BoundToExample()
            : base()
            {
                #region
                Initialization

                Credentials.ProjectName =
                "BoundToExample";
                #endregion
            }

            public override
        void Init()
            {

            }

            public override
        void Update(TickStatus args)
            {
                if (args ==
        TickStatus.IsBar)
                {
                    var orders
                    =
                    OrdersManager.GetOrders(false)
                    ;

                    if(orders.Count > 0)

                    orders.ForEach(order =>
                        {
                            if
                        (order.Side == OrderSide.Buy)
                            {

                                var updateRequest = new
                                ReplaceOrderRequest(order);
                            }
                        }
                    )
                }
            }
        }
```

```
if(updateRequest.BoundTo !=  
null)  
  
updateRequest.BoundTo = null;  
  
OrdersManager.UpdateAsync(upda  
teRequest, (result)=>{  
  
if (result.HasError)  
  
Notification.Comment($"  
{result.Message}");  
  
} );  
}  
}  
}  
}  
}
```

See Also

Reference

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

ReplaceOrder RequestComment Property

Comment for the order.

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public string Comment { get;  
    set; }
```

[Copy](#)

Property Value
[String](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
CommentExample :  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
CommentExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"CommentExample";
    #endregion
}

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(false)
;

if(orders.Count > 0)

orders.ForEach(order =>
{
    if
(order.Side == OrderSide.Buy)
{

var updateRequest = new
ReplaceOrderRequest(order);

updateRequest.Comment = $"An
```

```
order is update based on
{this.Credentials.ProjectName}
strategy";  
  
OrdersManager.UpdateAsync(updateRequest, (result)=>{  
  
    if (result.HasError)  
  
        Notification.Comment($"  
{result.Message}");  
  
    } );  
}  
}  
}  
}
```

See Also

Reference

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

ReplaceOrder RequestExpiration Time Property

The date of expiration order (for GTD orders).

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public DateTime?  
ExpirationTime { get; set; }
```

[Copy](#)

Property Value
[NullableDateTime](#)

► Example

C#

```
using System;  
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
ExpirationTimeExample :  
StrategyBuilder  
{
```

[Copy](#)

```
        public  
ExpirationTimeExample()  
        : base()  
{  
    #region  
Initialization  
  
Credentials.ProjectName =  
"ExpirationTimeExample";  
    #endregion  
}  
  
        public override  
void Init()  
{  
  
}  
  
        public override  
void Update(TickStatus args)  
{  
    if (args ==  
TickStatus.IsBar)  
    {  
        var orders =  
OrdersManager.GetOrders(false)  
;  
  
if(orders.Count > 0)  
  
orders.ForEach(order =>  
{  
    if  
(order.Side == OrderSide.Buy)  
    {  
  
var updateRequest = new  
ReplaceOrderRequest(order);  
  
updateRequest.ExpirationTime =
```

See Also

Reference

ReplaceOrderRequest Class

TradeApi.Trading Namespace

ReplaceOrder RequestOrder Property

Modifying order

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public Order Order { get; }
```

[Copy](#)

Property Value
Order

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class ReplaceOrderRequestExample :
        StrategyBuilder
    {
        public
ReplaceOrderRequestExample()
```

[Copy](#)

```
        : base()
    {
        #region
        Initialization

        Credentials.ProjectName =
        "ReplaceOrderRequestExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(false)
;

if(orders.Count > 0)

orders.ForEach(order =>
{
    var updateRequest = new
ReplaceOrderRequest(order);

Notification.Comment($"
{updateRequest.Order.Name} ");
}

OrdersManager.UpdateAsync(updateRequest, (result)=>{
```

```
    if (result.HasError)

        Notification.Comment($"
{result.Message}");

    } );
}
}
```

See Also

Reference

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

ReplaceOrder RequestPrice Property

Order's price

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double? Price { get;  
    set; }
```

[Copy](#)

Property Value
[NullableDouble](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
PriceExample : StrategyBuilder  
    {  
        public  
PriceExample()  
            : base()  
        {
```

[Copy](#)

```
        #region
Initialization

Credentials.ProjectName =
"PriceExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;

if(orders.Count > 0)

orders.ForEach(order =>
{
    if
(order.Side == OrderSide.Buy)
{

var price =
order.SLTPPriceType ==
SLTPPriceType.Absolute ?

order.Price + 50 *
InstrumentsManager.Current.Min
imalTickSize : 50;
```

```
var updateRequest = new  
ReplaceOrderRequest(order);  
  
updateRequest.Price = price;  
  
OrdersManager.UpdateAsync(updateRequest, (result) => {  
  
    if (result.HasError)  
  
        Notification.Comment($"  
{result.Message}") ;  
  
    } ) ;  
}  
}  
}  
}  
}
```

See Also

Reference

ReplaceOrderRequest Class

TradeApi.Trading Namespace

ReplaceOrder RequestQuantity Property

Value of the order in lots.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double? Quantity { get;  
    set; }
```

[Copy](#)

Property Value

[NullableDouble](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
QuantityExample :  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
QuantityExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"QuantityExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(false)
;

if(orders.Count > 0)

orders.ForEach(order =>
{
    if
(order.Side == OrderSide.Buy)
{

var updateRequest = new
ReplaceOrderRequest(order);

updateRequest.Quantity = 2;
}
```

```
OrdersManager.UpdateAsync(updateRequest, (result) => {

    if (result.HasError)

        Notification.Comment($""
{result.Message}"");

    }

});
```

See Also

Reference

ReplaceOrderRequest Class

TradeApi.Trading Namespace

ReplaceOrder RequestSLTPPriceType Property

Manages take profit/stop loss and trailing stop loss offset type as one of options: absolute or tick

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public SLTPPriceType?  
SLTPPriceType { get; set; }
```

[Copy](#)

Property Value
[NullableSLTPPriceType](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
SLTPPriceTypeExample:  
StrategyBuilder
```

[Copy](#)

```
    {
        public
SLTPPriceTypeExample()
        : base()
        {
            #region
Initialization

Credentials.ProjectName =
"SLTPPriceTypeExample";
            #endregion
        }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;
        if
(orders.Count > 0)

orders.ForEach(order =>
{
    if (order.Side ==
OrderSide.Buy)

{
```

```
var updateRequest = new  
ReplaceOrderRequest(order);  
  
updateRequest.SLTPPriceType =  
SLTPPriceType.Absolute;  
  
updateRequest.TakeProfitPrice  
=  
updateRequest.TakeProfitPrice  
+ 50 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
updateRequest.StopLossPrice =  
updateRequest.StopLossPrice +  
50 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
OrdersManager.UpdateAsync(updateRequest, (result) => {  
  
    if (result.HasError)  
  
        Notification.Comment($"  
{result.Message}");  
  
    } );  
}  
}  
}  
}
```

See Also

Reference

ReplaceOrderRequest Class
TradeApi.Trading Namespace

ReplaceOrder RequestStopLossLimit Price Property

Set price level of the stop loss limit order accompanied by primary order

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double?  
StopLossLimitPrice { get; set;  
}
```

[Copy](#)

Property Value
[NullableDouble](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
StopLossLimitPriceExample :  
StrategyBuilder
```

[Copy](#)

```
        {
            public
StopLossLimitPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"StopLossPriceLimtiExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;
        if
(orders.Count > 0)

orders.ForEach(order =>
{
}

if (order.Side ==
OrderSide.Buy)

{
```

```
var updateRequest = new  
ReplaceOrderRequest(order);  
  
updateRequest.SLTPPriceType =  
SLTPPriceType.Absolute;  
  
updateRequest.StopLossPrice =  
updateRequest.StopLossPrice +  
10 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
updateRequest.StopLossLimitPri  
ce =  
updateRequest.StopLossPrice +  
20 *  
InstrumentsManager.Current.Min  
imalTickSize;  
  
OrdersManager.UpdateAsync(upda  
teRequest, (result) => {  
  
if (result.HasError)  
  
Notification.Comment($"  
{result.Message}");  
  
});  
}  
});  
}  
}  
}
```

See Also

Reference

ReplaceOrderRequest Class
TradeApi.Trading Namespace

ReplaceOrder RequestStopLossPrice Property

Set price level of the stop loss order
accompanied by primary order

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double? StopLossPrice { Copy
    get; set; }
```

Property Value
[NullableDouble](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
StopLossPriceExample : StrategyBuilder
{
```

```
        public
StopLossPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"StopLossPriceExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(false)
;
    if
(orders.Count > 0)

orders.ForEach(order =>
{
}

if (order.Side ==
OrderSide.Buy)

{
var updateRequest = new
```

```
ReplaceOrderRequest(order);  
  
updateRequest.SLTPPriceType =  
    SLTPPriceType.Absolute;  
  
updateRequest.StopLossPrice =  
    updateRequest.StopLossPrice +  
    10 *  
    InstrumentsManager.Current.Min  
    imalTickSize;  
  
OrdersManager.UpdateAsync(updateRequest, (result) => {  
  
    if (result.HasError)  
  
        Notification.Comment($"  
        {result.Message}");  
  
    } );  
  
}  
}  
}  
}  
}  
}
```

See Also

Reference

ReplaceOrderRequest Class

TradeApi.Trading Namespace

ReplaceOrder RequestStopPrice Property

Stop Price for the StopLimit order

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double? StopPrice {  
    get; set; }
```

[Copy](#)

Property Value

[NullableDouble](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
StopPriceExample :  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
StopPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"StopPriceExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(false)
;

if(orders.Count > 0)

orders.ForEach(order =>
{
}

if(order.Side ==
OrderSide.Buy)
{

var price =
order.SLTPPriceType ==
SLTPPriceType.Absolute ?
```

```
InstrumentsManager.Current.Day  
Info.Ask - 100 *  
InstrumentsManager.Current.Min  
imalTickSize : -100;  
  
var updateRequest = new  
ReplaceOrderRequest(order);  
  
updateRequest.StopPrice =  
price;  
  
OrdersManager.UpdateAsync(updateRequest, (result)=>{  
  
if (result.HasError)  
  
Notification.Comment($"  
{result.Message}");  
  
});  
}  
}  
}  
}  
}
```

◀ ▶

• **See Also**

Reference

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

ReplaceOrder RequestTakeProfitPrice Property

Set price level of take profit order
accompanied by primary order

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public double? TakeProfitPrice Copy
{ get; set; }
```

Property Value
[NullableDouble](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
TakeProfitPriceExample:
StrategyBuilder
{
```

```
        public
TakeProfitPriceExample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TakeProfitPriceExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;
        if
(orders.Count > 0)

orders.ForEach(order =>
{
}

if (order.Side ==
OrderSide.Buy)

{
    var updateRequest = new
```

See Also

Reference

ReplaceOrderRequest Class

TradeApi.Trading Namespace

ReplaceOrder RequestTIF Property

Order's Time-in-force.

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public TimeInForce? TIF { get;  
    set; }
```

[Copy](#)

Property Value
[NullableTimeInForce](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
TimeInForceExample:  
StrategyBuilder  
    {  
        public  
TimeInForceExample()  
            : base()
```

[Copy](#)

```
        {
            #region
Initialization

Credentials.ProjectName =
"TimeInForceExample";
            #endregion
        }

    public override
void Init()
{
}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{
    var orders
=
OrdersManager.GetOrders(false)
;

if(orders.Count > 0)

orders.ForEach(order =>
{
    if
(order.Side == OrderSide.Buy)
{

var updateRequest = new
ReplaceOrderRequest(order);

updateRequest.TIF =
TimeInForce.GTC;

```

```
OrdersManager.UpdateAsync(updateRequest, (result) => {

    if (result.HasError)

        Notification.Comment($""
{result.Message}");

    } );
}

} ) ;

}
}
```

See Also

Reference

ReplaceOrderRequest Class

TradeApi.Trading Namespace

ReplaceOrder RequestTrailingStop Offset Property

Offset points to trigger the trailing stop loss.

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public double?  
TrailingStopOffset { get; set;  
}
```

[Copy](#)

Property Value
[NullableDouble](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
TrailingStopOffsetExample:  
StrategyBuilder  
{
```

[Copy](#)

```
        public
TrailingStopOffsetxample()
    : base()
{
    #region
Initialization

Credentials.ProjectName =
"TrailingStopOffsetxample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var orders
=
OrdersManager.GetOrders(false)
;
        if
(orders.Count > 0)

orders.ForEach(order =>
{
    if (order.Side ==
OrderSide.Buy)
    {

var updateRequest = new
ReplaceOrderRequest(order);

```

```
updateRequest.SLTPPriceType =  
    SLTPPriceType.Absolute;  
  
updateRequest.TrailingStopOffs  
    et =  
        updateRequest.TrailingStopOffs  
    et + 20 *  
        InstrumentsManager.Current.Min  
    imalTickSize;  
  
OrdersManager.UpdateAsync(updateRequest, (result) => {  
  
    if (result.HasError)  
  
        Notification.Comment($"  
            {result.Message}");  
  
    } );  
};  
}  
}  
}
```

See Also

Reference

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

ReplaceOrderRequest Methods

The [ReplaceOrderRequest](#) type exposes the following members.

▲ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	Finalize	(Overrides ObjectFinalize)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a

string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

ReplaceOrder RequestFinalize Method

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
protected override void  
Finalize()
```

[Copy](#)

Implements
[ObjectFinalize](#)

► See Also

[Reference](#)

[ReplaceOrderRequest Class](#)

[TradeApi.Trading Namespace](#)

SLTPPriceType Enumeration

SI/Tp price type

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum SLTPPriceType
```

[Copy](#)

► Members

Member name	Value	Description
Absolute	0	Absolute
Ticks	1	Ticks

► See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

StrategyBuilder Class

StrategyBuilder entity

► Inheritance Hierarchy

```
SystemObject  ScriptBase
CSharpScript
CSharpIndicator
TradeApi.ToolBeltBaseScript
TradeApi.TradingStrategyBuilder
```

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll) Version: 1.0.0

► Syntax

C#

```
public class StrategyBuilder : BaseScript,
    ICSharpStrategy, ICSharpIndicator,
    ICSharpScript, IDisposable
```

[Copy](#)

The [StrategyBuilder](#) type exposes the following members.

► Constructors

Name	Description
 StrategyBuilder	Constructor

[Top](#)

► Properties

Name	Description
 AccountManager	A manager reference to retrieve accounts (Inherited from BaseScript)
 Connection	Connection entity (Inherited from BaseScript)
 Credentials	Script credentials (Inherited from CSharpScript)
 CustomTradingHours	Custom working hours of the strategy
 Disposed	(Inherited from CSharpScript)
 HistoricalDataManager	A historicaldata manager reference to wield custom control over extra

		required historical element (Inherited from BaseScript)
 	HistoryDataSeries	Represent access to instrument's historical data based on current chart (Inherited from BaseScript)
 	Id	(Inherited from CSharpScript)
 	IndicatorsManager	An indicator manager reference to obtain usage of buildin and custom indicators (Inherited from BaseScript)
 	InstrumentsManager	A manager reference to query and manage vendor's instruments (Inherited from BaseScript)
 	Notification	An indicator's notification hub (Inherited from BaseScript)
 	OrdersManager	Provides manager for all orders
 	PositionsManager	Provides manager for all positions
 	ProjectType	(Inherited from CSharpScript)
 	SeparateWindow	(Inherited from CSharpIndicator)
 	SingleThreadEvents	(Inherited from CSharpScript)
 	Statistic	Statistic entity (Inherited from BaseScript)
 	Tools	Tool assistance in platform (Inherited from BaseScript)

[Top](#)

Methods

Name	Description
 	Complete This function is called before r script (Inherited from BaseScript)
 	Dispose (Inherited from CSharpScript)
 	Equals Determines whether the specif is equal to the current object. (Inherited from Object)
 	Finalize Allows an object to try to free

and perform other cleanup operations before it is reclaimed by garbage collection.
(Inherited from [Object](#))

≡	GetArray(ArrayList)	(Inherited from ScriptBase)
≡	GetArray(ArrayList, Boolean)	(Inherited from ScriptBase)
≡	GetArray(Type, Int32)	(Inherited from ScriptBase)
≡	GetHashCode	Serves as the default hash function. (Inherited from Object)
≡	GetType	Gets the Type of the current instance. (Inherited from Object)
≡ ≡	Init	This function is called once before the work starts. (Inherited from BaseScript)
≡	LoadAndInvoke	(Inherited from CSharpScript)
≡	MemberwiseClone	Creates a shallow copy of the Object . (Inherited from Object)
≡	OnDebug	(Inherited from CSharpScript)
≡	OnPaintChart	(Inherited from CSharpScript)
≡	PushMethodDump	(Inherited from CSharpScript)
≡	QueueUserWorkItem(WaitCallback)	(Inherited from CSharpScript)
≡	QueueUserWorkItem(WaitCallback, Object)	(Inherited from CSharpScript)
≡	QueueUserWorkItem(WaitCallback, Object, ApartmentState)	(Inherited from CSharpScript)
≡	RegisterLocalVariable(String, FuncObject, String)	(Inherited from CSharpScript)
≡	RegisterLocalVariable(String, FuncObject, ActionObject, String)	(Inherited from CSharpScript)
≡	ToString	Returns a string that represents the current object. (Inherited from Object)
≡ ≡	Update	This function is called amid new changes.

new bar or history processing.
argument with a tick status to
quote state
(args.TickStatus,IsHistory/IsB
(Inherited from [BaseScript](#))

[Top](#)

Fields

Name	Description
 currentInternalInstrument	(Inherited from BaseScript)
 ScriptCache	(Inherited from CSharpScript)

[Top](#)

See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

StrategyBuilder Constructor

Constructor

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public StrategyBuilder()
```

[Copy](#)

► See Also

[Reference](#)

[StrategyBuilder Class](#)

[TradeApi.Trading Namespace](#)

StrategyBuilder Properties

The [StrategyBuilder](#) type exposes the following members.

Properties

Name	Description
 AccountManager	A manager reference to retrieve accounts (Inherited from BaseScript)
 Connection	Connection entity (Inherited from BaseScript)
 Credentials	Script credentials (Inherited from CSharpScript)
 CustomTradingHours	Custom working hours of the strategy
 Disposed	(Inherited from CSharpScript)
 HistoricalDataManager	A historicaldata manager reference to wield custom control over extra

		required historical element (Inherited from BaseScript)
	HistoryDataSeries	Represent access to instrument's historical data based on current chart (Inherited from BaseScript)
	Id	(Inherited from CSharpScript)
	IndicatorsManager	An indicator manager reference to obtain usage of buildin and custom indicators (Inherited from BaseScript)
	InstrumentsManager	A manager reference to query and manage vendor's instruments (Inherited from BaseScript)
	Notification	An indicator's notification hub (Inherited from BaseScript)
	OrdersManager	Provides manager for all orders



PositionsManager

Provides manager
for all positions



ProjectType

(Inherited from
CSharpScript)



SeparateWindow

(Inherited from
CSharpIndicator)



SingleThreadEvents

(Inherited from
CSharpScript)



Statistic

Statistic entity
(Inherited from
BaseScript)



Tools

Tool assistance in
platform
(Inherited from
BaseScript)

[Top](#)

See Also

[Reference](#)

[StrategyBuilder Class](#)

[TradeApi.Trading Namespace](#)

StrategyBuilderCustomTradingHours Property

Custom working hours of the strategy

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string  
CustomTradingHours { get; set;  
}
```

[Copy](#)

Property Value
[String](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
using System;  
  
namespace TestEnv  
{  
    public class  
CustomTradingHoursExample :  
StrategyBuilder
```

[Copy](#)

```
{  
    public  
CustomTradingHoursExample()  
        : base()  
    {  
        #region  
Initialization  
  
Credentials.ProjectName =  
"CustomTradingHoursExample";  
        #endregion  
    }  
  
[InputParameter(InputType.Date  
Time, "Custom trading hours",  
1)]  
    public DateTime  
tradingHours;  
  
    public override  
void Init()  
    {  
  
this.CustomTradingHours =  
tradingHours.ToString("yyyy-MM-ddTHH:mm:ss")  
    }  
  
    public override  
void Update(TickStatus args)  
    {  
        if  
(DateTime.Parse(this.CustomTra  
dingHours) < DateTime.UtcNow)  
            return;  
    }  
}
```

See Also

[Reference](#)

[StrategyBuilder Class](#)

[TradeApi.Trading Namespace](#)

StrategyBuilderOrdersManager Property

Provides manager for all orders

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public OrdersManager  
OrdersManager { get; }
```

[Copy](#)

Property Value
[OrdersManager](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
OrdersManagerExample :  
StrategyBuilder  
{  
    public  
OrdersManagerExample()  
        : base()
```

[Copy](#)

```
        {
            #region
            Initialization

            Credentials.ProjectName =
                "OrdersManagerExample";
            #endregion
        }

        public override
void Init()
{
}

public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {

        Notification.Print($"Total
sell:
{OrdersManager.TotalSellQuantiti
ty}");

        Notification.Print($"Total
buy:
{OrdersManager.TotalBuyQuantit
y}");
    }
}
}
```

See Also

Reference

StrategyBuilder Class
TradeApi.Trading Namespace

Strategy BuilderPositions Manager Property

Provides manager for all positions

Namespace: [TradeApi.Trading](#)
Assembly: TradeApi (in TradeApi.dll)
Version: 1.0.0

► Syntax

C#

```
public PositionsManager  
PositionsManager { get; }
```

[Copy](#)

Property Value
[PositionsManager](#)

► Example

C#

```
using Runtime.Script;  
using TradeApi.Trading;  
  
namespace TestEnv  
{  
    public class  
PositionsManagerExample :  
StrategyBuilder  
{  
    public
```

[Copy](#)

```
PositionsManagerExample()
    : base()
{
    #region
    Initialization

    Credentials.ProjectName =
    "PositionsManagerExample";
    #endregion
}

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
{

    Notification.Print($"Total
sell position:
{PositionsManager.GetPositions
(position => position.Side ==
PositionSide.Short)}");

    Notification.Print($"Total buy
position:
{PositionsManager.GetPositions
(position => position.Side ==
PositionSide.Long)}");
}
}
```

See Also

[Reference](#)

[StrategyBuilder Class](#)

[TradeApi.Trading Namespace](#)

StrategyBuilder Methods

The [StrategyBuilder](#) type exposes the following members.

Methods

Name	Description
 Complete	This function is called before running the script (Inherited from BaseScript)
 Dispose	(Inherited from CSharpScript)
 Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 Finalize	Allows an object to try to free up memory and perform other cleanup operations before it is reclaimed by garbage collection. (Inherited from Object)
 GetArray(ArrayList)	(Inherited from ScriptBase)
 GetArray(ArrayList, Boolean)	(Inherited from ScriptBase)
 GetArray(Type, Int32)	(Inherited from ScriptBase)
 GetHashCode	Serves as the default hash function. (Inherited from Object)
 GetType	Gets the Type of the current instance. (Inherited from Object)
 Init	This function is called once before the work starts (Inherited from BaseScript)
 LoadAndInvoke	(Inherited from CSharpScript)
 MemberwiseClone	Creates a shallow copy of the current Object . (Inherited from Object)
 OnDebug	(Inherited from CSharpScript)
 OnPaintChart	(Inherited from CSharpScript)

💡	PushMethodDump	(Inherited from CSharpScript)
💡	QueueUserWorkItem(WaitCallback)	(Inherited from CSharpScript)
💡	QueueUserWorkItem(WaitCallback, Object)	(Inherited from CSharpScript)
💡	QueueUserWorkItem(WaitCallback, Object, ApartmentState)	(Inherited from CSharpScript)
💡	RegisterLocalVariable(String, FuncObject, String)	(Inherited from CSharpScript)
💡	RegisterLocalVariable(String, FuncObject, ActionObject, String)	(Inherited from CSharpScript)
💡	ToString	Returns a string that represents the current object. (Inherited from Object)
💡 ⚡	Update	This function is called amid new bar or history processing. argument with a tick status to quote state (args.TickStatus,IsHistory/IsB) (Inherited from BaseScript)

[Top](#)

See Also

[Reference](#)

[StrategyBuilder Class](#)

[TradeApi.Trading Namespace](#)

StrategyBuilder Fields

The [StrategyBuilder](#) type exposes the following members.

↳ Fields

	Name	Description
💡	currentInternalInstrument	(Inherited from BaseScript)
💡	ScriptCache	(Inherited from CSharpScript)

[Top](#)

↳ See Also

[Reference](#)

[StrategyBuilder Class](#)

[TradeApi.Trading Namespace](#)

TimeInForce Enumeration

TimeInForce

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public enum TimeInForce
```

[Copy](#)

► Members

Member name	Value	Description
GTC	10,008	GTC
IOC	10,010	IOC
Day	10,011	Day
FOK	11,000	FOK
GTD	11,003	GTD
GTS	11,008	GTS

► See Also

Reference

TradeApi.Trading Namespace

TradeResult Class

TradeResult entity

▪ Inheritance Hierarchy

[SystemObject](#) [TradeApi.TradingTradeResult](#)

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

▪ Syntax

C#

```
public sealed class  
TradeResult
```

[Copy](#)

The [TradeResult](#) type exposes the following members.

▪ Constructors

	Name	Description
≡	TradeResult	Initializes a new instance of the TradeResult class

[Top](#)

▪ Properties

	Name	Description
 	HasError	Determines either order has been placed right or wrong with signaling an error existence
 	Id	Returns an id number if order has been placed
 	Message	Returns an error message if order is not valid

[Top](#)

◀ Methods

	Name	Description
 	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
 	GetHashCode	Serves as the default hash function. (Inherited from Object)
 	GetType	Gets the Type of the current instance.

(Inherited from
[Object](#))



[ToString](#)

Returns a string that represents the current object.
(Inherited from [Object](#))

[Top](#)

◀ See Also

[Reference](#)

[TradeApi.Trading Namespace](#)

TradeResult Constructor

Initializes a new instance of the [TradeResult](#) class

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

◀ Syntax

C#

```
public TradeResult()
```

[Copy](#)

◀ See Also

[Reference](#)

[TradeResult Class](#)

[TradeApi.Trading Namespace](#)

TradeResult Properties

The [TradeResult](#) type exposes the following members.

► Properties

	Name	Description
 	HasError	Determines either order has been placed right or wrong with signaling an error existence
 	Id	Returns an id number if order has been placed
 	Message	Returns an error message if order is not valid

[Top](#)

► See Also

[Reference](#)

[TradeResult Class](#)

[TradeApi.Trading Namespace](#)

TradeResultHasError Property

Determines either order has been placed right or wrong with signaling an error existence

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public bool HasError { get; } Copy
```

Property Value

Boolean

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
HasErrorExample :
StrategyBuilder
{
    public
HasErrorExample()
```

```
        : base()
    {
        #region Initialization

        Credentials.ProjectName =
"HasErrorExample";
        #endregion
    }

        public override
void Init()
{
}

        public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
newRequest = new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);
        var result =
OrdersManager.Send(newRequest)
;

    if(result.HasError)

        Notification.Print($"Order
place error:
{result.Message}");
    }
}
```

```
        }  
    }
```

See Also

Reference

[TradeResult Class](#)

[TradeApi.Trading Namespace](#)

TradeResultId Property

Returns an id number if order has been placed

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Id { get; }Copy
```

Property Value

String

► Example

C#

```
using Runtime.Script;Copy
using TradeApi.Trading;

namespace TestEnv
{
    public class IdExample
        : StrategyBuilder
    {
        public IdExample()
            : base()
        {
            #region
```



```
        }  
    );  
}  
}  
}
```

See Also

Reference

[TradeResult Class](#)

[TradeApi.Trading Namespace](#)

TradeResultMessage Property

Returns an error message if order is not valid

Namespace: [TradeApi.Trading](#)

Assembly: TradeApi (in TradeApi.dll)

Version: 1.0.0

► Syntax

C#

```
public string Message { get; } Copy
```

Property Value

[String](#)

► Example

C#

```
using Runtime.Script;
using TradeApi.Trading;

namespace TestEnv
{
    public class
ErrorExample : StrategyBuilder
    {
        public
ErrorExample()
            : base()
        {
            #region
```

```
Initialization

Credentials.ProjectName =
"ErrorExample";
                                #endregion
}

    public override
void Init()
{

}

    public override
void Update(TickStatus args)
{
    if (args ==
TickStatus.IsBar)
    {
        var
newRequest = new
OrderRequest(OrderType.Limit,
InstrumentsManager.Current,
AccountManager.Current,
OrderSide.Buy, 1);
        var result =
OrdersManager.Send(newRequest)
;

if(result.HasError)

Notification.Print($"Order
place error:
{result.Message}");
    }
}
}
```

See Also

[Reference](#)

[TradeResult Class](#)

[TradeApi.Trading Namespace](#)

TradeResult Methods

The [TradeResult](#) type exposes the following members.

▪ Methods

	Name	Description
	Equals	Determines whether the specified object is equal to the current object. (Inherited from Object)
	GetHashCode	Serves as the default hash function. (Inherited from Object)
	GetType	Gets the Type of the current instance. (Inherited from Object)
	ToString	Returns a string that represents the current object. (Inherited from Object)

[Top](#)

◀ See Also

[Reference](#)

[TradeResult Class](#)

[TradeApi.Trading Namespace](#)
