

1	Introduction .....	2
2	IVolatility SDK Interfaces Diagram.....	3
3	IVDDSEnt interface. ....	5
3.1	Initialization.....	5
3.2	Accessing Stock data.....	5
3.3	Accessing Options data. ....	6
3.3.1	Single option contract.....	6
3.3.2	All options contracts for selected underlying.....	6
3.3.3	All options contracts for selected underlying and selected strike. ....	7
3.3.4	All options contracts for selected underlying and selected expiration. ....	7
3.3.5	All available options contracts. ....	7
3.4	Accessing IV Index data.....	8
3.4.1	IV Indexes for selected underlying.....	8
3.4.2	IV Indexes for selected underlying and term (IV index horizon) .....	8
3.4.3	IV Indexes for all available instruments.....	8
3.5	Accessing IV Surface data.....	9
3.5.1	IV Surface data for selected underlying .....	9
3.5.2	IV Surface data for selected underlying and selected surface term (surface horizon) 9	
3.5.3	IV Surface data for selected underlying and selected moneyness.....	10
3.5.4	IV Surface data for all instruments.....	10
3.5.5	Single IV Surface data point.....	10
3.6	Accessing Historical Volatility (HV) data.....	11
3.7	Accessing Interest Rate data .....	11
4	IStocksEnt Interface .....	12
5	IOptionsEnt interface.....	15
6	IVIndexesEnt interface .....	17
7	IVSurfacesEnt interface.....	18
8	HVEnt interface.....	19
9	InterestRateEnt interface .....	19
10	Subscription mechanism.....	21
11	DDE Server .....	23
11.1	Requesting Stock data .....	23
11.2	Requesting Options data.....	24
11.3	Requesting IVIndex data .....	24
11.4	Requesting IVSurface data .....	25
12	Appendix 1 .....	26

# 1 Introduction

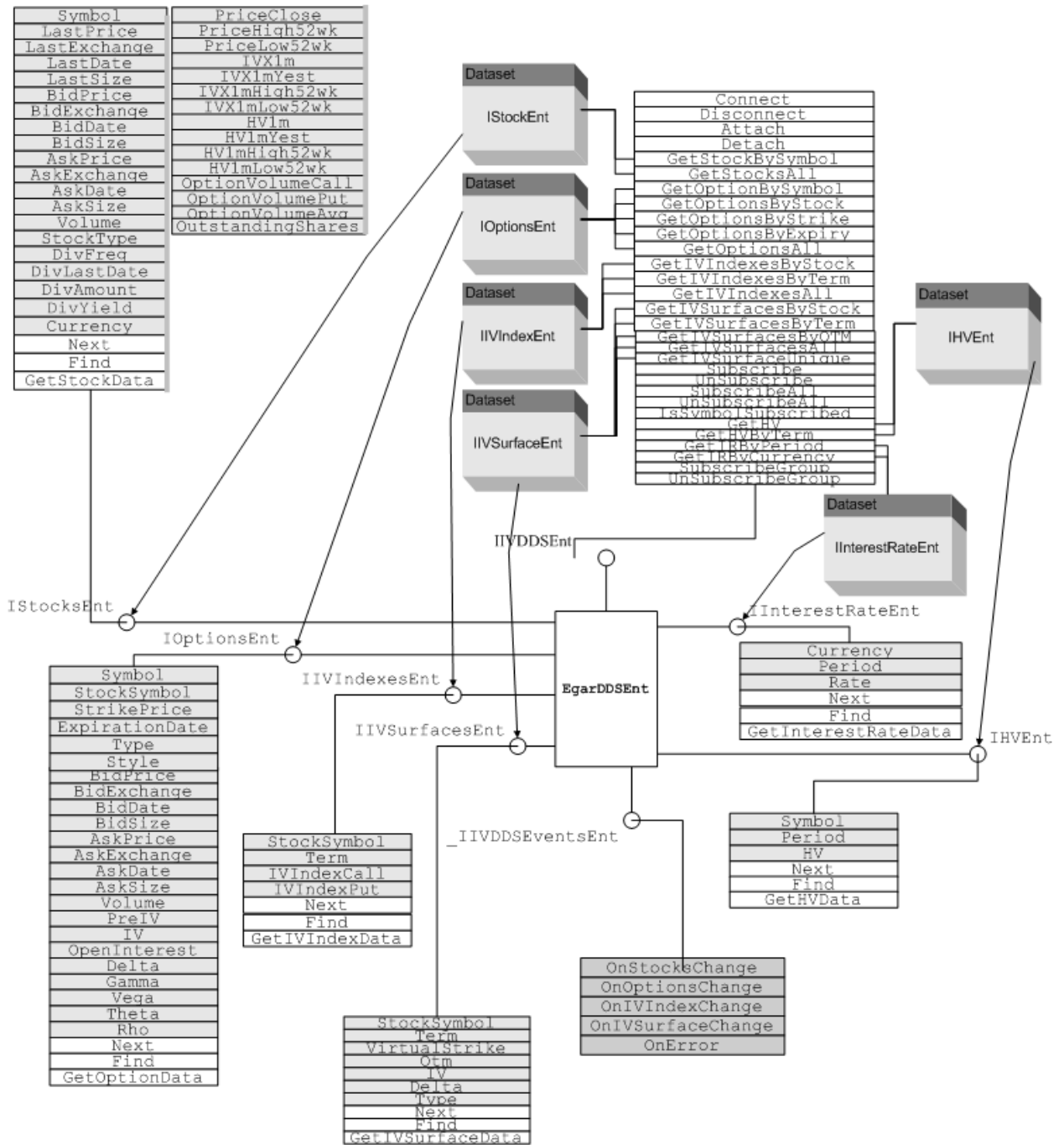
The IVolatility SDK is a set of software libraries providing simple COM interface access to U.S. stock and options quotes and derivatives data such as implied volatilities, Greeks, volatility surface etc. With industry standard COM interface you get an unprecedented flexibility with feeding real-time quotes and derivatives data into your applications. Now you can build your own risk-management software at a very low cost and in a fraction of usual development time. The SDK also provides a DDE link to Excel, so you can use dynamically updating data in your Excel applications.

The SDK offers both event-driven and request based interfaces for flexible data access. Please refer to the following chapters for a complete SDK class diagram and in-depth description.

The IVolatility SDK consists of several class interfaces that will be described in greater details below. The core interface **IVDDSEnt** is used to initialize the system (to establish connection and perform authorization on IVolatility.com master data server). Upon successful object creation you should first do a `Connect()` request to perform initial connection and user authorization. Then you can either query the data using any lookup method we support or subscribe to receive data updates. For example you can get all option contracts for the specified underlying stock using `getOptionsbyStock()` method. You can also subscribe for real-time data updates using one of several subscription methods we offer (`SubscribeAll()` method, for example, will subscribe you to all instruments you are authorized to access). Upon successful subscription a pre-defined call-back method (`onOptionsChange` method to receive individual option contracts quotes and volatilities changes) will be called every time the value is updated.

IVolatility SDK also includes DDE server so you can access real-time volatility data in Microsoft Excel.

## 2 IVolatility SDK Interfaces Diagram



IVolatility SDK Interfaces diagram

“DDS Enterprise Library” is a core ActiveX object of the IVolatility SDK. It provides two primary interfaces to work with.

First interface, IIVDDSEnt, is the main interface to get access to the data. It has methods to initialize the system (Connect, Disconnect) and several methods to retrieve the data and subscribe to data updates. All data is stored in collections. There are 5 types of data collection – one per each data type: IStocksEnt, IOptionsEnt, IIInterestRateEnt, IIVIndexesEnt and IIVSurfacesEnt. A collection of specified data type can be accessed using such methods as GetStockBySymbol, GetStocksAll, GetOptionBySymbol etc.

Each data collection is a reference to local 'in-memory' database and has an internal pointer to 'current' record. Methods 'Next' and 'Find' allows iterating over data collection records. Method 'Next' moves internal pointer to the next record. Method 'Find' positions internal pointer to the first record matching requested stock or index symbol.

Data collection interfaces have unique set of properties. Each 'property' can be accessed selectively or you can get entire set by using such methods as GetStockData.

Second interface, IIVDDSEventsEnt interface gives you ability to receive notifications about data updates in real-time. To start receiving updates you should call 'Subscribe...' method first. The SubscribeAll method allows subscribing to receive all 'data changed' notifications on all instruments your client is permitted to. After successful subscription EgarDDSEnt object will start sending notifications your application using On...Change methods. So your application should implement these methods. Each 'update' event will contain IStocksEnt, IOptionsEnt, IIVIndexesEnt or IIVSurfacesEnt data collection with 'updated' dataset.

## 3 IVDDSEnt interface.

### 3.1 Initialization

Method Connect() is used to establish connection with IVolatility master data source. You should always call this method first.

*Definition:*

```
HRESULT Connect([in] BSTR UserName, [in] BSTR Password);
```

*Parameters:*

UserName – username

Password – password

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password
```

The Disconnect() method is used to disconnect from master data source. You should always call this method before closing your application.

*Definition:*

```
HRESULT Disconnect();
```

*Parameters:*

No parameters

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
...  
dds.Disconnect
```

### 3.2 Accessing Stock data

Method GetStockBySymbol and GetStocksAll returns a Stocks collection. See IStock interface description below to see how to work with Stocks collection.

*Definition:*

```
HRESULT GetStockBySymbol([in] BSTR Symbol, [out,retval] IStocks** pStocks);  
HRESULT GetStocksAll([out, retval] IStocks **pStocks);
```

*Parameters:*

Symbol – ticker symbol.

pStocks – IStock collection interface

*Example 1:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
Dim stck As StocksEnt  
Set stck = dds. GetStockBySymbol (Symbol)  
...  
dds.Disconnect
```

*Example 2:*

```
Dim dds As New IVDDSEnt
dds.Connect username, password
Dim stcksCol As StocksEnt
Set stcksCol = dds.GetStocksAll
....
dds.Disconnect
```

### **3.3 Accessing Options data.**

Options data can be initialized in a collection object. You can fill in this collection by only one option contract or by a group of option contracts. There are several methods that allow you to fill Options collection with:

#### **3.3.1 Single option contract**

*Definition:*

```
HRESULT GetOptionBySymbol ([in] BSTR Symbol, [out, retval] IOptions** pOptions);
```

*Parameters:*

Symbol – option symbol  
pData – IOptions collection reference

*Example:*

```
Dim dds As New IVDDSEnt
dds.Connect username, password
Dim optn As OptionsEnt
Set optn = dds. GetOptionBySymbol (“IBMDA”)

dds.Disconnect
```

#### **3.3.2 All options contracts for selected underlying**

*Definition:*

```
HRESULT GetOptionsByStock([in] BSTR Symbol, [out, retval] IOptions** pOptions);
```

*Parameters:*

Symbol – Stock symbol  
pOptions – IOptions collection reference

*Example:*

```
Dim dds As New IVDDSEnt
dds.Connect username, password
Dim optnsCollection As OptionsEnt
Set optnsCollection = dds. GetOptionsByStock (Symbol)
...
dds.Disconnect
```

### 3.3.3 All options contracts for selected underlying and selected strike.

*Definition:*

```
HRESULT GetOptionsByStrike([in] BSTR StockSymbol, [in] DOUBLE StrikePrice, [out, retval] IOptions** pOptions);
```

*Parameters:*

StockSymbol – Stock symbol.  
StrikePrice – option Strike price.  
pOptions – IOptions collection reference

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
Dim optnsCollection As OptionsEnt  
Set optnsCollection = dds.GetOptionsByStrike (StockSymbol, StrikePrice)  
....  
dds.Disconnect
```

### 3.3.4 All options contracts for selected underlying and selected expiration.

*Definition:*

```
HRESULT GetOptionsByExpiry([in] BSTR StockSymbol, [in] DATE ExpirationDate, [out, retval] IOptions** pOptions);
```

*Parameters:*

StockSymbol – Stock symbol.  
ExpirationDate – option expiration date  
pOptions – IOptions collection reference

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
Dim optnsCollection As OptionsEnt  
Set optnsCollection = dds.GetOptionsByExpiry (StockSymbol, ExpirationDate)  
....  
dds.Disconnect
```

### 3.3.5 All available options contracts.

This method returns a collection of all options that specified client is authorized to receive.

*Definition:*

```
HRESULT GetOptionsAll([out, retval] IOptions** pOptions);
```

*Parameters:*

pOptions – IOptions collection reference

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
Dim optnsCollection As OptionsEnt  
Set optnsCollection = dds.GetOptionsAll  
dds.Detach enOptionData  
....
```

dds.Disconnect

### 3.4 Accessing IV Index data.

IV Index data is also can be initialized into a collection. Following methods are available.

#### 3.4.1 IV Indexes for selected underlying.

*Definition:*

```
HRESULT GetIVIndexesByStock([in] BSTR StockSymbol, [out, retval] IIVIndexes** pIVIndexes);
```

*Parameters:*

StockSymbol – underlying symbol  
pIVIndexes – IVIndexes collection reference

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
Dim ivxCollection As IVIndexesEnt  
Set ivxCollection = dds. GetIVIndexesByStock(StockSymbol)  
...  
dds.Disconnect
```

#### 3.4.2 IV Indexes for selected underlying and term (IV index horizon)

*Definition:*

```
HRESULT GetIVIndexesByTerm([in] BSTR StockSymbol, [in] SHORT Term, [out, retval]  
IIVIndexData** pIVIndexData);
```

*Parameters:*

StockSymbol – underlying symbol  
Term – term (in calendar days)  
pIVIndexes – IVIndexes collection reference

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
Dim ivxCollection As IVIndexesEnt  
Set ivxCollection = dds. GetIVIndexesByTerm (StockSymbol, Term)  
...  
dds.Disconnect
```

#### 3.4.3 IV Indexes for all available instruments

GetIVIndexesAll returns a collection of IV Indexes for all instruments current user is authorized to receive.

*Definition:*

```
HRESULT GetIVIndexesAll([out, retval] IIVIndexes** pIVIndexes);
```

*Parameters:*

pIVIndexes – IVIndexes collection reference

*Example:*

```
Dim dds As New IVDDSEnt
dds.Connect username, password
Dim ivxCollection As IVIndexesEnt
Set ivxCollection = dds. GetIVIndexesAll
...
dds.Disconnect
```

### **3.5 Accessing IV Surface data.**

Implied volatility surface data is initialized in a collection object. There are several methods that allow you to fill in IVSurface collection.

#### **3.5.1 IV Surface data for selected underlying**

*Definition:*

```
HRESULT GetIVSurfacesByStock([in] BSTR StockSymbol, [out, retval] IIVSurfaces** pIVSurfaces);
```

*Parameters:*

StockSymbol – underlying symbol.  
pIVSurfaces – IVSurfaces collection reference

*Example:*

```
Dim dds As New IVDDSEnt
dds.Connect username, password
Dim ivsCollection As IVSurfaces
Set ivsCollection = dds.GetIVSurfacesByStock(StockSymbol)
...
dds.Detach enIVIndexData
dds.Disconnect
```

#### **3.5.2 IV Surface data for selected underlying and selected surface term (surface horizon)**

*Definition:*

```
HRESULT GetIVSurfacesByTerm([in] BSTR StockSymbol, [in] SHORT Term, [out, retval] IIVSurfaces** pIVSurfaces);
```

*Parameters:*

StockSymbol – underlying symbol  
Term – term (in calendar days)  
pIVSurfaces – IVSurfaces collection reference

*Example:*

```
Dim dds As New IVDDSEnt
dds.Connect username, password
Dim ivsCollection As IVSurfacesEnt
Set ivsCollection = dds. GetIVSurfacesByTerm(StockSymbol, Term)
...
dds.Disconnect
```

### 3.5.3 IV Surface data for selected underlying and selected moneyness.

*Definition:*

```
HRESULT GetIVSurfacesByOTM([in] BSTR StockSymbol, [in] SHORT OTM, [out, retval] IIVSurfaces** pIVSurfaces);
```

*Parameters:*

StockSymbol – underlying symbol  
OTM - moneyness  
pIVSurfaces – IVSurfaces collection reference

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
Dim ivsCollection As IVSurfacesEnt  
Set ivsCollection = dds. GetIVSurfacesByOTM (StockSymbol, OTM)
```

### 3.5.4 IV Surface data for all instruments

GetIVSurfacesAll returns a collection of IV Surface records for all instruments current user is authorized to receive.

*Definition:*

```
HRESULT GetIVIndexesAll([out, retval] IIVSurfaces** pIVSurfaces);
```

*Parameters:*

pIVIndexes – IVSurfaces collection reference

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password  
Dim ivsCollection As IVSurfacesEnt  
Set ivsCollection = dds. GetIVSurfacesAll  
...  
dds.Disconnect
```

### 3.5.5 Single IV Surface data point

Method GetIVSurfaceUnique returns a single volatility point on the Volatility Surface curve for selected instrument and specified term, moneyness (OTM value) and Call/Put curve portion (Call for virtual strikes above spot, Put for strikes below spot)

*Definition:*

```
HRESULT GetIVSurfaceUnique([in] BSTR StockSymbol, [in] SHORT Term, [in] SHORT OTM, [in] OptionTypeEnum Type, [out, retval] IIVSurfaces** pIVSurfaceData)
```

*Parameters:*

StockSymbol – underlying symbol  
Term – term (days)  
OTM – moneyness  
Type – enCall or enPut  
pIVSurfaces – IVSurfaces collection reference

*Example:*

```
Dim dds As New IVDDSEnt  
dds.Connect username, password
```

```

Dim ivs As IVSurfacesEnt
Set ivs = dds. GetIVSurfaceUnique(StockSymbol, Term, OTM, Type)
...
dds.Disconnect

```

### 3.6 Accesing Historical Volatility (HV) data

Method GetHV returns HV data collection for given symbol, GetHVByTerm returns HV value for given symbol and term. See HVEnt interface description below to see how to work with HV data collection.

*Definition:*

```

HRESULT GetHV([in] BSTR Symbol, [out, retval] IHVEnt** pHV);
HRESULT GetHVByTerm([in] BSTR Symbol, [in] SHORT Term, [out, retval]
FLOAT* HV);

```

*Parameters:*

Symbol – security symbol.  
Period - term

*Example:*

```

Dim dds As New IVDDSEnt
Dim HV30 As Float
dds.Connect username, password
dds.GetHVByTerm("MSFT", 30, HV30)
...
dds.Disconnect

```

### 3.7 Accesing Interest Rate data

Method GetIRByPeriod returns IR value for given currency and period, GetIRByCurrency returns Interest rate data collection. See InterestRateEnt interface description below to see how to work with Interest rate collection.

*Definition:*

```

HRESULT GetIRByPeriod([in] BSTR Currency, [in] LONG Period, [out, retval] FLOAT* Rate);
HRESULT GetIRByCurrency([in] BSTR Currency, [out, retval] IInterestRateEnt** pInterestRate);

```

*Parameters:*

Currency – currency symbol.  
Period - period

*Example:*

```

Dim dds As New IVDDSEnt
Dim rate As Float
dds.Connect username, password
dds.GetIRByPeriod("USD", 60, rate)
...
dds.Disconnect

```

## 4 IStocksEnt Interface

IStocks interface provides a set of methods to iterate over Stock data collection and access each item properties.

Symbol	HRESULT Symbol([out, retval] BSTR *pVal) Stock Symbol
LastPrice	HRESULT LastPrice([out, retval] FLOAT *pVal) LastPrice
LastExchange	HRESULT LastExchange([out, retval] BSTR *pVal) Exchange of last price
LastDate	HRESULT LastDate([out, retval] DATE *pVal) Time or date of last price
LastSize	HRESULT LastSize([out, retval] LONG *pVal) Last size
BidPrice	HRESULT BidPrice([out, retval] FLOAT *pVal) Bid Price
BidExchange	HRESULT BidExchange([out, retval] BSTR *pVal) Exchange of bid price
BidDate	HRESULT BidDate([out, retval] DATE *pVal) Time or date of bid price
BidSize	HRESULT BidSize([out, retval] LONG *pVal) Bid size
AskPrice	HRESULT AskPrice([out, retval] FLOAT *pVal) Ask Price
AskExchange	HRESULT AskExchange([out, retval] BSTR *pVal) Exchange of ask price
AskDate	HRESULT AskDate([out, retval] DATE *pVal) Time or date of ask price
AskSize	HRESULT AskSize([out, retval] LONG *pVal) Ask size
Volume	HRESULT Volume([out, retval] long *pVal) Volume
StockType	HRESULT StockType([out, retval] StockTypeEnum *pVal) enum StockTypeEnum <pre>         {             enIndex                = 0,             enStock                 = 1,         } StockTypeEnum; </pre> Returns security type.
PriceClose	HRESULT PriceClose([out, retval] FLOAT *pVal); Close price as of yesterday
PriceHigh52wk	HRESULT PriceHigh52wk([out, retval] FLOAT *pVal); Highest price for the last year
PriceLow52wk	HRESULT PriceLow52wk([out, retval] FLOAT *pVal); Lowest price for the last year

IVX1m	HRESULT IVX1m([out, retval] FLOAT *pVal); 1-m IV Index as of yesterday close
IVX1mYest	HRESULT IVX1mYest([out, retval] FLOAT *pVal); Day before yesterdays value of IV Index
IVX1mHigh52wk	HRESULT IVX1mHigh52wk([out, retval] FLOAT *pVal); Highest value of 1m IV Index for the last year
IVX1mLow52wk	HRESULT IVX1mLow52wk([out, retval] FLOAT *pVal); Lowest value of 1m IV Index for the last year
HV1m	HRESULT HV1m([out, retval] FLOAT *pVal); Historical Volatility (HV) 1m as of yesterday
HV1mYest	HRESULT HV1mYest([out, retval] FLOAT *pVal); Day before yesterdays value of HV 1m
HV1mHigh52wk	HRESULT HV1mHigh52wk([out, retval] FLOAT *pVal); Highest value of HV 1m for the last year
HV1mLow52wk	HRESULT HV1mLow52wk([out, retval] FLOAT *pVal); Lowest value of HV 1m for the last year
OptionVolumeCall	HRESULT OptionVolumeCall([out, retval] long *pVal); Total Calls Volume
OptionVolumePut	HRESULT OptionVolumePut([out, retval] long *pVal); Total Puts Volume
OptionVolumeAvg	HRESULT OptionVolumeAvg([out, retval] FLOAT *pVal); One month average of total Calls and Puts Volume
OutstandingShares	HRESULT OutstandingShares([out, retval] FLOAT *pVal); Outstanding Shares
DivFreq	HRESULT DivFreq([out, retval] long *pVal) Dividend frequency code: 4 - Quarterly 2 - Semi-annually 1 - Annually 0 - No dividends
DivLastDate	HRESULT DivLastDate([out, retval] DATE *pVal) Last dividend date
DivAmount	HRESULT DivAmount([out, retval] FLOAT *pVal) Dividend amount or yield
Currency	HRESULT Currency([out, retval] BSTR *pVal) Currency
Next	HRESULT Next([out,retval] VARIANT_BOOL* bVal) Move to the next elemnt in a collection. Last element will return VARIANT_FALSE as bVal value
Find	HRESULT Find([in] BSTR Symbol, [out,retval] VARIANT_BOOL* bVal) Find symbol in collection

*Example:*

```
Dim stck As StocksEnt
Set stck = dds.GetStocksAll
Do
Debug.Print stck.Symbol & " " & stck.LastPrice
```

Loop While stck.Next

## 5 IOptionsEnt interface

IOptionsEnt interface provides a set of methods to iterate over Options dataset collection and access each item properties.

Symbol	HRESULT StockSymbol([out, retval] BSTR*pVal); Stock symbol
StockSymbol	HRESULT StockSymbol([out, retval] BSTR *pVal) Underlying symbol
OptionStyle	HRESULT OptionStyle([out, retval] OptionStyleEnum *pVal) enum OptionStyleEnum { enAmerican = 0, enEuropean = 1 } OptionStyleEnum; Option style American or European
BidPrice	HRESULT BidPrice([out, retval] FLOAT *pVal) Bid Price
BidExchange	HRESULT BidExchange([out, retval] BSTR *pVal) Exchange of bid price
BidDate	HRESULT BidDate([out, retval] DATE *pVal) Time or date of bid price
BidSize	HRESULT BidSize([out, retval] LONG *pVal) Bid size
AskPrice	HRESULT AskPrice([out, retval] FLOAT *pVal) Ask Price
AskExchange	HRESULT AskExchange([out, retval] BSTR *pVal) Exchange of ask price
AskDate	HRESULT AskDate([out, retval] DATE *pVal) Time or date of ask price
AskSize	HRESULT AskSize([out, retval] LONG *pVal) Ask size
StrikePrice	HRESULT StrikePrice([out, retval] float *pVal); Strike price
ExpirationDate	HRESULT ExpirationDate([out, retval] DATE *pVal); Expiration date
IV	HRESULT IV([out, retval] float *pVal); Implied Volatility (interpolated value)
PreIV	HRESULT IV([out, retval] float *pVal); Implied Volatility (not interpolated value)
Delta	HRESULT Delta([out, retval] float *pVal); Delta
Gamma	HRESULT Gamma([out, retval] float *pVal); Gamma
Vega	HRESULT Vega([out, retval] float *pVal); Vega

Theta	HRESULT Theta([out, retval] float *pVal); Theta
Rho	HRESULT Rho([out, retval] float *pVal); Rho
Volume	HRESULT Volume([out, retval] long *pVal); Volume
OpenInterest	HRESULT OpenInterest([out, retval] long *pVal); Open Interest
GetOptionData	HRESULT GetOptionData([out] BSTR* Symbol, [out] BSTR* StockSymbol, [out] OptionTypeEnum* OptionType, [out] float* StrikePrice, [out] DATE* ExpirationDate, [out] float* IV, [out] float* Bid, [out] float* Ask, [out] float* Delta, [out] float* Vega, [out] float* Gamma, [out] float* Theta, [out] float* Rho, [out] long* Volume); Get all Options properties in a single request.
Next	HRESULT Next([out,retval] VARIANT_BOOL* bVal) Move to the next elemnt in a collection. Last element will return VARIANT_FALSE as bVal value
Find	HRESULT Find([in] BSTR Symbol, [out,retval] VARIANT_BOOL* bVal)

*Example:*

```

Dim optn As IOptionEnt
Symbol = "MSFT"
Set optn = dds.GetOptionsByStock(Symbol)
Do
    Debug.Print optn.Symbol & " " & optn.IV
Loop While optn.Next

```

## 6 IVIndexesEnt interface

IVIndexes interface provides a set of methods to iterate over data collection and access each item properties.

StockSymbol	HRESULT StockSymbol([out, retval] BSTR *pVal) Stock Symbol
Term	HRESULT Term([out, retval] short *pVal) IVX term (days)
IVIndexCall	HRESULT IVIndexCall([out, retval] float *pVal) IVIndexCall – IV Index for Call options
IVIndexPut	HRESULT IVIndexPut([out, retval] float *pVal) IVIndexPut – IV Index for Put options
GetIVIndexData	HRESULT GetIVIndexData([out] BSTR* StockSymbol, [out] short* Term, [out] float* IVIndexCall, [out] float* IVIndexPut) Get all IVIndex properties in a single request.
Next	HRESULT Next([out,retval] VARIANT_BOOL* bVal) Move to the next element in a collection. Last element will return VARIANT_FALSE as bVal value
Find	HRESULT Find([in] BSTR Symbol, [out,retval] VARIANT_BOOL* bVal)

*Example:*

```
Dim ivxs As IVIndexes
Symbol = "MSFT"
Set ivxs = dds.GetIVIndexesByStock(Symbol)
Do
    Debug.Print ivxs.Symbol & " " & ivxs.Term & " " & ivxs.IVIndexCall
Loop While ivxs.Next
```

## 7 IVSurfacesEnt interface

IVSurfaces interface provides a set of methods to iterate over data collection and access each item properties.

StockSymbol	HRESULT StockSymbol([out, retval] BSTR *pVal) Stock symbol
Term	HRESULT Term([out, retval] short *pVal) IV Surface term (days)
VirtualStrike	VirtualStrike([out, retval] float *pVal)
Otm	HRESULT Otm([out, retval] short *pVal); OTM value
Type	HRESULT Type([out, retval] OptionTypeEnum *pVal) Type (enCall or enPut)
IV	HRESULT IV([out, retval] float *pVal) Implied Volatility
Delta	HRESULT Delta([out, retval] float *pVal) Delta
GetIVSurfaceData	HRESULT GetIVSurfaceData([out] BSTR* StockSymbol, [out] short* Term, [out] float* VirtualStrike, [out] short* Otm, [out] float* IV, [out] float* Delta, [out] OptionTypeEnum* Type) Get all IVSurfaces properties in a single request.
Next	HRESULT Next([out,retval] VARIANT_BOOL* bVal) Move to the next elemnt in a collection. Last element will return VARIANT_FALSE as bVal value
Find	HRESULT Find([in] BSTR Symbol, [out,retval] VARIANT_BOOL* bVal)

*Example:*

```
Dim ivs As IVSurfaces
Symbol = "MSFT"
Set ivs = dds.GetIVSurfacesByStock(Symbol)
Do
    Debug.Print ivs.StockSymbol & " " & ivs.Term & " " & ivs.Otm & " " & ivs.Type & " " &
    ivs.IV
Loop While ivs.Next
```

## 8 HVEnt interface

HVEnt interface provides a set of methods to iterate over data collection and access each item properties.

Symbol	HRESULT Symbol([out, retval] BSTR *pVal); Security symbol
Period	HRESULT Period([out, retval] LONG *pVal) Period
HV	HRESULT HV([out, retval] FLOAT *pVal); Historical Volatility value
GetHVDData	HRESULT GetHVDData([out] BSTR* Symbol, [out] SHORT* Period, [out] FLOAT* HV); Return all data available
Next	HRESULT Next([out,retval] VARIANT_BOOL* bVal) Move to next element
Find	HRESULT Find([in] BSTR Symbol, [out,retval] VARIANT_BOOL *bVal) Find record set by currency symbol

Example:

```
Dim hvr As HVEnt
Symbol = "MSFT"
Set hvr = dds.GetHV(Symbol)
Do
    Debug.Print hvr.Period & " " & hvr.HV & vbCRLF
Loop While ivs.Next
```

## 9 InterestRateEnt interface

InterestRateEnt interface provides a set of methods to iterate over data collection and access each item properties.

Currency	HRESULT Currency([out, retval] BSTR *pVal) Currency symbol
Period	HRESULT Period([out, retval] LONG *pVal) Period
Rate	HRESULT Rate([out, retval] FLOAT *pVal) Interest Rate value
GetInterestRateData	HRESULT GetInterestRateData([out] BSTR* Currency,[out] LONG* Period, [out] FLOAT* Rate); Return all data available
Next	HRESULT Next([out,retval] VARIANT_BOOL* bVal) Move to next element
Find	HRESULT Find([in] BSTR Symbol, [out,retval] VARIANT_BOOL *bVal) Find record set by currency symbol

Example:

```
Dim ivs As InterestRateEnt
Symbol = "USD"
Set ir = dds.GetIRByCurrency(Symbol)
Do
    Debug.Print ir.Period & " " & ir.Rate & vbCRLF
Loop While ivs.Next
```



## 10 Subscription mechanism

Data update subscription mechanism provides data update using event-driven model. You should enter security symbol, symbol type and dataset type (see Appendix 1).

Subscribe	HRESULT Subscribe([in] BSTR Symbol, [in] SymbolTypeEnum SymbType, [in] DataTypeEnum Type)  Subscribe to received updates for the specified symbol
UnSubscribe	HRESULT UnSubscribe([in] BSTR Symbol, [in] SymbolTypeEnum SymbType, [in] DataTypeEnum Type);  Cancel receiving updates for the specified symbol
SubscribeAll	HRESULT SubscribeAll([in] DataTypeEnum Type);  Subscribe to receive updates for all symbols available for current user
UnSubscribeAll	HRESULT UnSubscribeAll([in] DataTypeEnum Type);  Cancel receiving updates for all symbols available to the current user
SubscribeGroup	HRESULT SubscribeGroup(VARIANT symbolArray, SymbolTypeEnum SymbType, DataTypeEnum Type)  Subscribe to receive updates for a group of symbols
UnSubscribeGroup	HRESULT UnSubscribeGroup(VARIANT symbolArray, SymbolTypeEnum SymbType, DataTypeEnum Type)  Cancel receiving updates for a group of symbols
IsSymbolSubscribed	HRESULT IsSymbolSubscribed([in] BSTR Symbol, [in] SymbolTypeEnum SymbType, [in] DataTypeEnum Type, [out,retval] VARIANT_BOOL* bVal);  Check if the symbol subscription is activated – “True” returned if succeed

### Example

```
Dim WithEvents dds as IVDDSEnt
...
dds.Subscribe "MSFT", enEquity , enStockData
...
if IsSymbolSubscribed("MSFT", enEquity, enStockData) = true then
    dds.UnSubscribe "MSFT", enEquity, enStockData
```

To handle data events you should declare call back methods for each data type.

OnStocksChange	HRESULT OnStocksChange([in] IStocksEnt* pData); Stock or Index quote changed
OnOptionsChange	HRESULT OnOptionsChange([in] IOptionsEnt* pData); Options data changed
OnIVIndexChange	HRESULT OnIVIndexChange([in] IIVIndexesEnt* pData); IVIndex data changed
OnIVSurfaceChange	HRESULT OnIVSurfaceChange([in] IIVSurfacesEnt* pData); IVSurface data changed

OnError	HRESULT OnError([in] ErrorNumberEnum ErrorNumber, [in] BSTR Description); Internal error occurred while sending or receiving data
---------	--

Example:

```
Private Sub dds_OnStocksChange(ByVal stckData As StocksEnt)
    Do
        Debug.Print stck.Symbol & " " & stck.LastPrice
    Loop While stckData.Next
End Sub
```

## 11 DDE Server

General format for DDE requests is “=Service|Topic!Item” where:

- Service – service name (IVDDE)
- Topic – Stock or Option symbol
- Item – data type

See complete description below.

### 11.1 Requesting Stock data

Service	IVDDE
Topic	Stock Symbol
Item	<ul style="list-style-type: none"> <li>• StckPrice</li> <li>• StckVolume</li> <li>• StckLastTime</li> <li>• StckLastExch</li> <li>• StckLastSize</li> <li>• StckBid</li> <li>• StckAsk</li> <li>• StckBidTime</li> <li>• StckAskTime</li> <li>• StckBidExch</li> <li>• StckAskExch</li> <li>• StckBidSize</li> <li>• StckAskSize</li> <li>• DivDate</li> <li>• DivAmount</li> <li>• DivFreq</li> <li>• StckPriceClose</li> <li>• StckPriceHigh52wk</li> <li>• StckPriceLow52wk</li> <li>• Ivx1m</li> <li>• Ivx1mYest</li> <li>• Ivx1mHigh52wk</li> <li>• Ivx1mLow52wk</li> <li>• Hv1m</li> <li>• Hv1mYest</li> <li>• Hv1mHigh52wk</li> <li>• Hv1mLow52wk</li> <li>• StckOptnVolumeCall</li> <li>• StckOptnVolumePut</li> <li>• StckOptnVolumeAvg</li> <li>• OutstandingShares</li> </ul>

*Examples:*

=IVDDE|MSFT!StckPrice – last price for MSFT

=IVDDE|DELL!StckVolume – today volume for DELL

## 11.2 Requesting Options data

Service	IVDDE
Topic	Option symbol
Item	<ul style="list-style-type: none"> <li>• StockSymbol</li> <li>• Strike</li> <li>• Type</li> <li>• ExpDate</li> <li>• IV</li> <li>• OptnBidTime</li> <li>• OptnAskTime</li> <li>• OptnBidExch</li> <li>• OptnAskExch</li> <li>• OptnBidSize</li> <li>• OptnAskSize</li> <li>• PreIv</li> <li>• Bid</li> <li>• Ask</li> <li>• GrkDelta</li> <li>• GrkVega</li> <li>• GrkGamma</li> <li>• GrkTheta</li> <li>• GrkRho</li> <li>• OptnVolume</li> <li>• OpenInterest</li> </ul>

### Examples:

=IVDDE|MQFAO!StockSymbol – underlying for option contract MQFAO

=IVDDE|MQFAO!Strike –MQFAO strike

=IVDDE|MQFAO!Type – MQFAO type (Call or Put)

=IVDDE|MQFAO!ExpDate – MQFAO expiration date

=IVDDE|MQFAO!IV –MQFAO implied volatility

=IVDDE|MQFAO!GrkVega – MQFAO Vega

=IVDDE|MQFAO!GrkRho –MQFAO Rho

=IVDDE|MQFAO!OptnVolume – MQFAO daily volume

## 11.3 Requesting IVIndex data

Service	IVDDE
Topic	Stock symbol
Item	IVX[Type]_T[Term], where <ol style="list-style-type: none"> <li>1. [Type] – option type ( “Call” or “Put”)</li> <li>2. [Term] – term (days)</li> </ol>

### Examples:

=IVDDE|MSFT!IVXCall\_T30 – 30-day Calls IVIndex for MSFT

=IVDDE|MSFT!IVXPut\_T90 – 90-day Put IVIndex for MSFT

## 11.4 Requesting IVSurface data

Service	IVDDE
Topic	Stock symbol
Item	IVS[Type]_T[Term]O[OTM]_[Value], where <ol style="list-style-type: none"><li>1. [Type] – surface type (“Call”, “Put”)</li><li>2. [Term] – term</li><li>3. [OTM] – OTM</li><li>4. [Value] – one of these: “VStrike” (virtual strike), “IV”, “Delta”</li></ol>

*Examples:*

=IVDDE|MSFT!IVSCALL\_T30O50\_IV – 30-day IVSurface implied volatility for 50% Calls  
OTM  
=IVDDE|MSFT!IVSPUT\_T90O0\_VSTRIKE – virtual strike for 90-day 0 OTM Puts for stock  
MSFT

## 12 Appendix 1

```
enum DataTypeEnum
```

```
{  
    enStockData           = 0,  
    enOptionData          = 1,  
    enIVIndexData         = 2,  
    enIVSurfaceData       = 3,  
    enMarketStructure     = 4  
} DataTypeEnum;
```

```
enum OptionTypeEnum
```

```
{  
    enCall                 = 0,  
    enPut                  = 1  
} OptionTypeEnum;
```

```
enum SymbolTypeEnum
```

```
{  
    enEquity               = 1,  
    enOption               = 2  
} SymbolTypeEnum;
```