xingAPI COM 개발가이드

COM이란?

COM(Component Object Model) 은 마이크로소프트에서 내놓은 일종의 프로그래밍 표준입니다.
COM 은 이 기술을 지원하는 모든 언어에서 프로세스간 통신과 동적 오브젝트 생성을 가능하게 하기 위해 사용됩니다. 이 이야기는 COM을 지원하는 모든 언어들은 사용이 가능하다는 것을 뜻합니다.

사용하기가 편합니다.

xingAPI DLL 버전은 "가장 빠르고 가장 가볍게" 라는 목표로 개발되어졌습니다.
그렇기 때문에 많은 규약이나 기능들을 개발자의 능력으로 해결하게끔 되어있습니다.
하지만, COM 버전은 쉽게 만들 수 있게 하자는 목표로 개발되어졌습니다.
데이터를 보낼 때의 데이터 규약 이라던지 각종 사전 지식을 익히지 않아도 쉽게 배워서 쉽게 사용이 가능합니다.

구성객체

xingAPI COM 버전은 3개의 객체로 구성되어 있습니다.

객체명	설명	파일명
XASession	서버연결, 로그인 등	XA_Session.dll
XAQuery	조회TR	XA_DataSet.dll
XAReal	실시간TR	XA_DataSet.dll

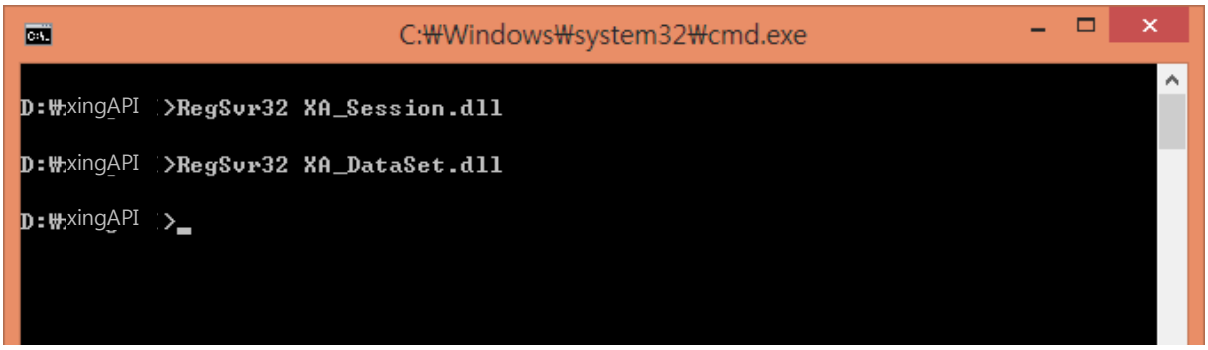
xingAPI DLL 을 기본으로 사용합니다.

xingAPI COM버전은 xingAPI DLL의 기능을 사용하고 덧붙여서 개발자들이 쉽게 사용하기 위한 함수를 추가하였습니다. 따라서, xingAPI COM버전을 사용하기 위해서는 xingAPI DLL 버전이 필요합니다.

COM 등록

COM버전은 xingAPI 홈페이지의 '최신버전 설치'를 통해 자동으로 등록되며, 이 부분은 COM버전을 수동으로 등록할 필요가 있을 때 참고하시면 됩니다

COM 버전을 사용하기 위해서는 레지스트리에 등록을 해야 합니다. 등록하는 프로그램은 RegSvr32.exe 입니다. RegSvr32.exe 는 윈도우에서 제공하는 프로그램으로 C:₩Windows₩System32 폴더 내에 있습니다.
사용방법은 "RegSvr32.exe 파일명" 입니다.
Command Prompt 에서 사용하셔야 하며, 등록할 파일은 XA_Session.dll, XA_DataSet.dll 입니다.
즉, Command Prompt 에서 RegSvr32 XA_Session.dll, RegSvr32 XA_DataSet.dll 를 입력하시면 됩니다.



위의 절차가 어려운 분들을 위해서 reg.bat 라는 파일을 넣어놨습니다.
Reg.bat 를 실행하시면 자동으로 등록이 됩니다.

※ Windows Vista 이상 Windows 7, Windows 8 인 경우에는 위와 같이 실행할 경우에는 에러가 발생을 합니다.

이 때에는 reg.bat 파일을 관리자 권한으로 실행하면 됩니다.
관리자 권한으로 실행하는 방법은

1. reg.bat 파일을 오른쪽 클릭하여
2. 나온 팝업메뉴에서 "관리자 권한으로 실행"을 누르면 됩니다.

하지만, 이럴 경우 파일이 없다고 에러가 발생할 수 있습니다.
이 때에는 reg.bat를 메모장으로 연후에 Path를 입력해주면 됩니다.
예를 들어, D:₩xingAPI 폴더에 파일이 있다고 한다면

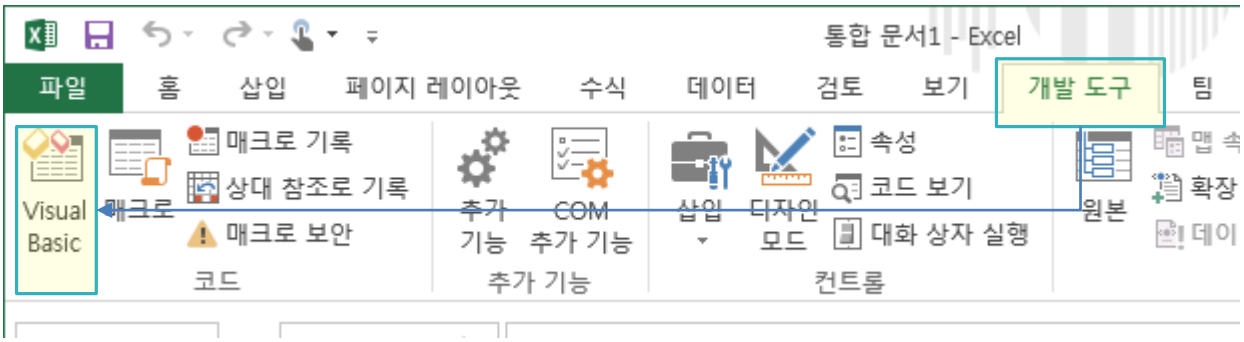
regsvr32 D:₩xingAPI₩XA_Session.dll
regsvr32 D:₩xingAPI₩XA_DataSet.dll

라고 입력하고 저장한 후에 관리자 권한으로 실행하면 됩니다.

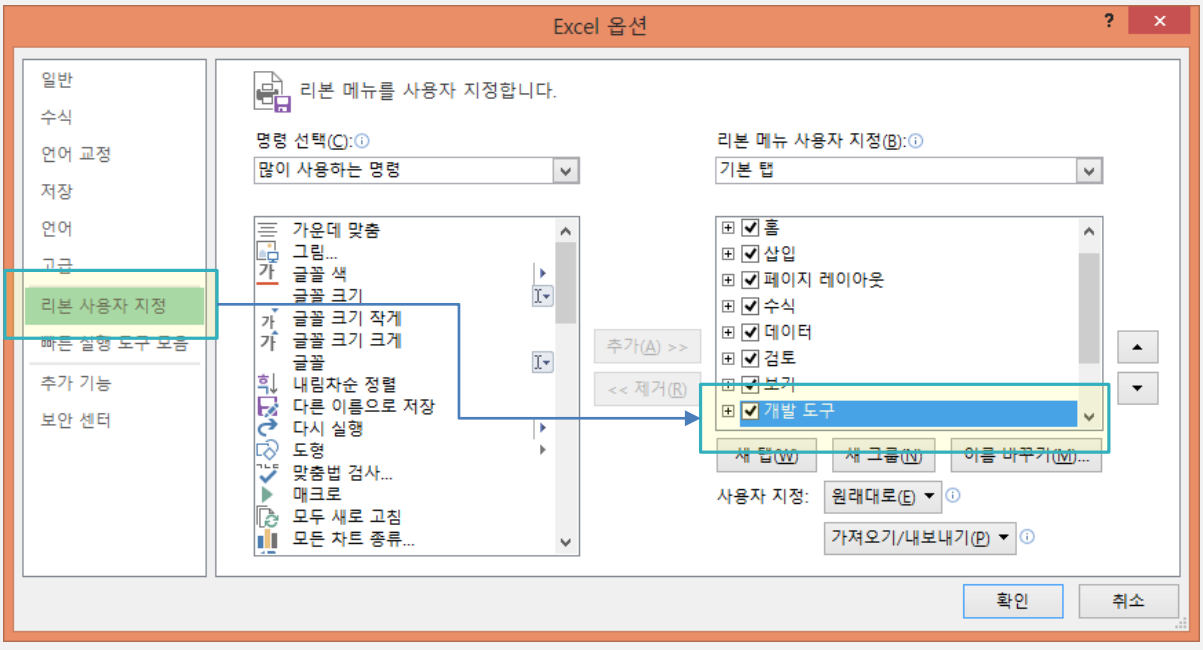
COM 명시

COM을 사용하기 위해서는 대부분의 언어에서는 사용한다고 명시를 해야 합니다.
이것은 언어마다 다르기 때문에 Microsoft Excel 2013을 기준으로 설명하도록 하겠습니다.
Microsoft Excel 2013 보다 낮은 버전도 대부분 대동소이 합니다.
(앞으로의 예제도 Excel 을 위주로 설명을 드리도록 하겠습니다.)

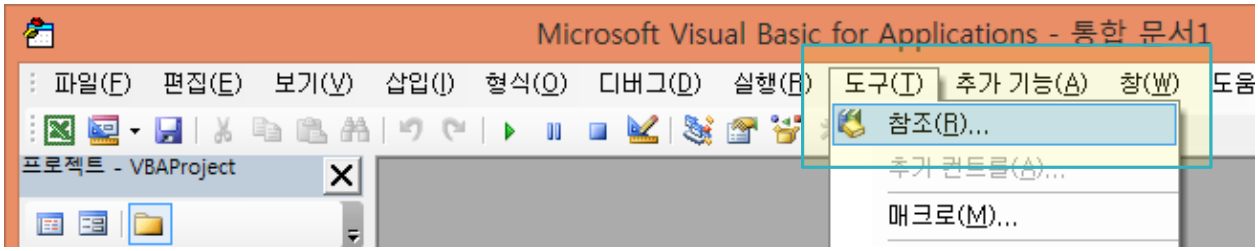
Step 1 메뉴에서 개발도구를 선택한 후에 Visual Basic 을 선택하여 Microsoft Visual Basic for Applications 를 실행합니다.



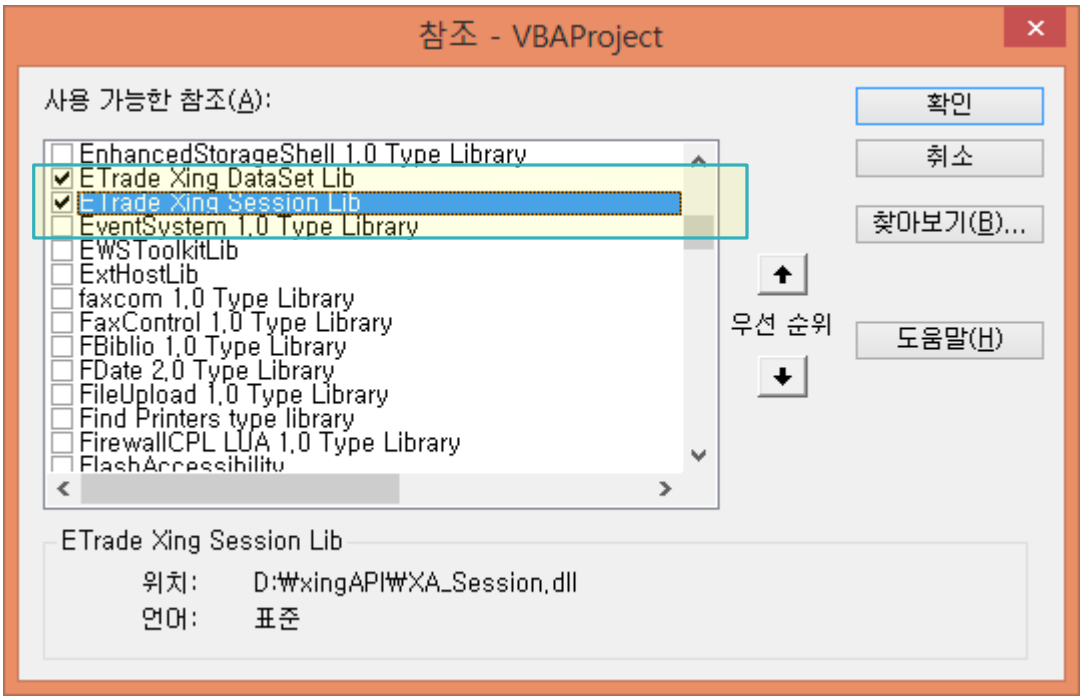
※ 개발도구 메뉴가 없을 경우에는 "Excel 옵션 > 리본 사용자 지정" 에서 개발 도구를 체크.



Step 2 Microsoft Visual Basic for Applications 의 메뉴에서 도구 > 참조 를 실행합니다.



Step 3 참조 다이얼로그에서 "Etrade Xing DataSetLib" 와 "Etrade Xing Session Lib" 를 체크하고 확인버튼을 누르면 명시가 완료됩니다.



서버 접속하기

xingAPI를 사용하기 위해서는 먼저 서버를 연결해야 합니다.
서버를 연결하기 위해서는 XASession 객체의 ConnectServer 메쏘드를 사용합니다.

Step 1 객체 선언하기

XASession 객체를 사용하기 위해서는 다음과 같이 선언합니다.

```
Dim WithEvents XASession_Excel As XASession
```

Step 2 객체 생성하기

선언을 한 후에 객체를 생성합니다.

```
Set XASession_Excel = CreateObject( "XA_Session.XASession" )
```

Step 3 서버 연결하기

객체를 생성한 후에 서버에 연결을 합니다.

```
bConnect = XASession_Excel.ConnectServer( "hts.ebestsec.co.kr", 20001 )
```

※ 모의투자의 경우, 접속주소는 "demo.ebestsec.co.kr" 입니다.

Step 4 에러체크하기

서버연결에 실패했을 경우에는 bConnect 에는 False 가 들어오게 됩니다.
이럴 경우에는 에러가 발생한 원인을 찾기 위해 GetLastError() 라는 함수를 사용하면 됩니다.

```
if bConnect = Fasle then
    nErrCode = XASession_Excel.GetLastError()
    strErrMsg = XASession_Excel.GetErrorMessage( nErrCode )
    MsgBox strErrMsg
End if
```

Full Code 서버에 접속하기 위한 Full Code 입니다.

```
' XASession의 변수를 선언한다.
Dim WithEvents XASession_Excel As XASession

Private Sub cbConnect_Click()

    ' XASession 객체를 생성한다.
    If XASession_Excel Is Nothing Then
        Set XASession_Excel = CreateObject( "XA_Session.XASession" )
    End If

    ' 이미 접속이 되어 있으면 접속을 끊는다.
    Call XASession_Excel.DisconnectServer

    ' 서버에 연결한다.
    If XASession_Excel.ConnectServer( "hts.ebestsec.co.kr", 20001 ) = False Then
        nErrCode = XASession_Excel.GetLastError()
        strErrMsg = XASession_Excel.GetErrorMessage( nErrCode )
        MsgBox strErrMsg
    End If

End Sub
```

서버 연결 끊김 이벤트

서버 연결 끊김 이벤트는 XASession 객체의 Disconnect 이벤트입니다.

DisconnectServer 메소드를 호출할 때 발생하는 이벤트가 아니며, 서버에서 강제로 연결을 끊는다는 통지의 의미도 아닙니다.

Disconnect이벤트는 OS의 소켓 끊김 이벤트 발생 시, 해당 이벤트를 그대로 받아 전달합니다.
그러나, 소켓이 끊기면 OS는 소켓 끊김을 자동으로 아는 것이 아니라 소켓에 어떤 action이 발생했을 때 (TR을 조회하는 등) 연결상태를 알게 되므로 Disconnect 이벤트가 바로 발생하지 않을 수도 있습니다. 따라서, 24시간 연결을 해야만 하는 경우에는 유의하여야 합니다.

```
Private Sub XASession_Excel_Disconnect()
    MsgBox "서버와의 연결이 끊겼습니다."
End Sub
```

로그인하기

서버에 접속이 되었다면 로그인을 해야 합니다.
로그인을 하지 않고는 어떠한 서비스도 사용이 불가능합니다.
로그인하기 위해서는 XASession 객체의 Login 메쏘드를 사용합니다.

Step 1 로그인하기

Login 메쏘드를 사용하여 로그인을 합니다.

```
bLogin = XASession_Excel.Login( 사용자아이디, 사용자비밀번호, 공인인증비밀번호, 0, False )
```

4번째 인자는 실서버일 경우에는 0, 모의투자서버일 경우에는 1을 입력하게 되어 있으나,
현재는 자동으로 체크하게끔 변경되었으므로 0을 입력하시면 됩니다.
5번째 인자는 공인인증 에러 시, 에러 메시지창을 표시할지 여부입니다.

Step 2 로그인 에러 체크하기

로그인시에 서버가 연결되지 않거나 혹은 전송에러가 발생할 경우 False를 반환합니다.
즉, False를 반환했을 경우에는 로그인정보가 서버로 전송하지 못할 경우에만 발생하며
서버로 전송한 후에 발생한 에러(비밀번호 오류등)는 Login 이벤트로 에러가 전송됩니다.

```
if bLogin = Fasle then
    MsgBox "로그인 서버전송에 실패하였습니다."
End if
```

Step 3 로그인 결과 받기

로그인이 성공적으로 서버로 전송이 되면 로그인에 대한 결과값은 Login 이벤트로 들어오게 됩니다.

```
Private Sub XASession_Excel_Login( ByVal szCode As String, ByVal szMsg As String )
    if szCode = "0000" then
        MsgBox "로그인에 성공했습니다."
    else
        MsgBox ""로그인 실패 : " & "[" & szCode & "] " & szMsg
    end if
End Sub
```

※ szCode 에 "0000" 이 들어올 경우는 로그인 성공이고 그 이외의 값은 로그인 실패입니다.

Full Code **로그인을 하기 위한 Full Code 입니다.**

```
Private Sub cbLogin_Click()

    If XASession_Excel.Login( szUserID, szUsePwd, szCertPwd, 0, False ) = False Then
        MsgBox "로그인 처리에 실패하였습니다.
    End If

End Sub

Private Sub XASession_Excel_Login(ByVal szCode As String, ByVal szMsg As String)

     if szCode = "0000" then
        MsgBox "로그인에 성공했습니다."
    else
        MsgBox ""로그인 실패 : " & "[" & szCode & "] " & szMsg
    end if

End Sub
```

계좌 가져오기

Step 1 계좌 개수 가져오기

먼저 계좌의 개수를 가져와야 합니다.

계좌의 개수를 가져오는 메쏘드는 GetAccountListCount() 입니다.

```
nAccountCount = XASession_Excel.GetAccountListCount()
```

Step 2 계좌리스트 가져오기

계좌의 개수를 알아낸 후에 계좌의 개수만큼 반복적으로 계좌를 가져옵니다.

```
For i=0 To nAccountCount – 1
    szAccount = XASession_Excel.GetAccountList( i )
Next
```

Full Code 계좌 가져오기 Full Code 입니다.

```
nAccountCount = XASession_Excel.GetAccountListCount()

For i=0 To nAccountCount – 1
    szAccount = XASession_Excel.GetAccountList( i )
Next
```

일반 데이터 조회

xingAPI에서의 데이터 조회는 미리 정해진 형식의 입력값을 서버로 전송하고 서버로부터 데이터를 받는 것을 의미합니다.
한 번의 전송으로 한번의 결과값을 받습니다.
현재가를 예로 들면 종목코드를 서버로 전송하면 그 당시의 현재가를 결과값으로 서버로부터 데이터를 받게 되는 것입니다.

1) TR (Transaction)

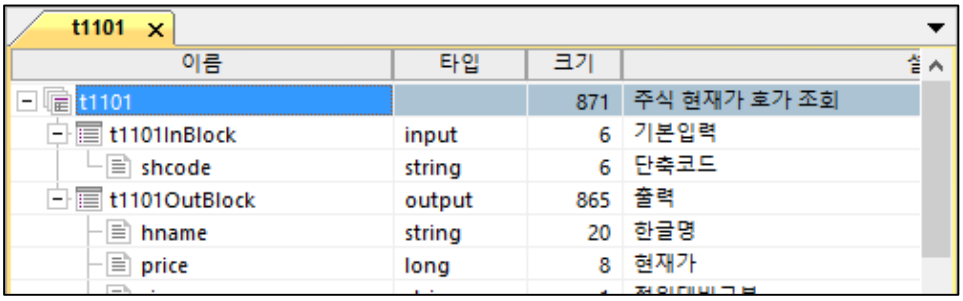
xingAPI에서의 Transaction은 서버로부터 데이터를 주고받는 행위를 일컫습니다. 종목코드를 입력해서 현재가를 받아오는 것도 TR이고 계좌번호를 입력해서 잔고를 받아오는 것도 TR입니다.

- TR Code

입력값으로 종목코드를 입력했을 경우 서버에서 종목코드만으로는 현재가 데이터를 주어야 할지 분별 데이터를 주어야 할지 모르기 때문에 그것들을 구별할 수 있도록 TR코드를 같이 입력해 주어야 합니다. TR코드는 5자리인 경우와 10자리인 경우가 있습니다. 예를 들어, t1101 은 현재가 등의 종목시세 데이터를 가져오는 TR코드입니다.

- TR Layout

데이터 조회는 미리 정의된 입력값들을 입력한 후에 서버로 전송하고 서버에서 입력값들에 의해 데이터를 처리한 후에 미리 정의된 형식으로 결과값을 전송합니다. 이런 미리 정의된 입력값과 결과값들을 TR Layout 이라고 부르며 DevCenter를 이용하여 확인할 수 있습니다.



DevCenter에 있는 [t1101] 주식 현재가 호가 조회입니다. t1101는 TR코드이며 주식 현재가 호가 조회는 TR명입니다.

TR Layout에는 크게 InBlock 과 OutBlock 이 있습니다. InBlock은 입력값이며 OutBlock은 출력값이고, InBlock과 OutBlock은 여러 개가 올 수 있습니다.

- RES

RES는 TR Layout 을 COM 버전에서 인식할 수 있는 형식으로 변경한 구조를 의미합니다.
COM버전은 RES를 입력해 주어야 동작이 가능합니다.
RES파일은 TR Layout을 파일로 저장한 것을 의미합니다.

단일 데이터 조회

서버에 연결하고 로그인을 하였다면, 이제 서버로부터 데이터를 가져오는 것을 할 수가 있습니다.
데이터를 가져오기 위해서는 XAQuery 객체를 사용하시면 됩니다.
XAQeury 객체 하나당 하나의 TR Code를 등록할 수가 있습니다.

XAQuery 생성 > Res 등록 > 데이터 입력 > 요청 > 받기 > 데이터 가져오기 순으로 사용하면 됩니다.

Step 1 XAQuery 선언

```
Dim WithEvents XAQuery_t1101 As XAQuery
```

Step 2 XAQuery 생성

```
Set XAQuery_t1101 = CreateObject( "XA_DataSet.XAQuery" )
```

Step 3 RES 등록

- 원하는 TR코드에 대한 RES 파일을 DevCenter에서 다운로드.
- 특정폴더에 복사

※ 참고
COM버전은 다음 순서로 RES 폴더를 검사합니다.
해당 폴더에 없을 경우에는 읽어들이지 못합니다.
① 실행파일이 위치한 폴더
② COM 파일이 위치한 폴더

- RES파일을 ResFileName 을 이용하여 XAQuery 객체에 등록

```
XAQuery_t1101.ResFileName = "t1101.res"
```

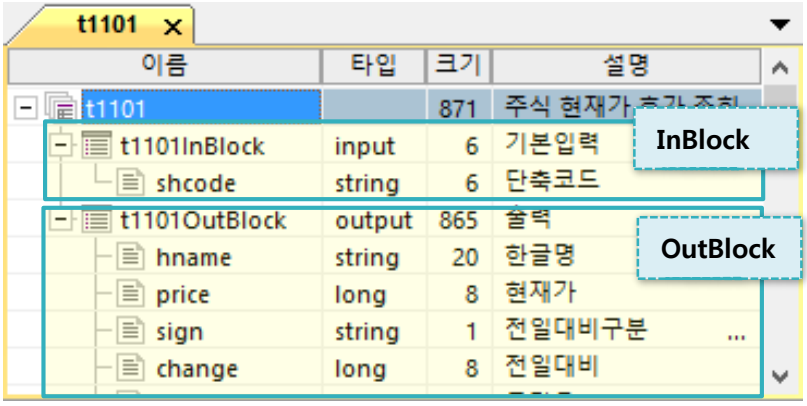
- 실행파일 혹은 COM파일이 위치한 폴더가 아닐 경우에는 Full Path를 입력하면 됩니다.
- Excel은 Property로 디자인타임 시간에는 RES파일을 등록하는 것이 불가능 하지만 다른 언어는 Property를 이용하여 디자인 타임 시간에 등록이 가능합니다.

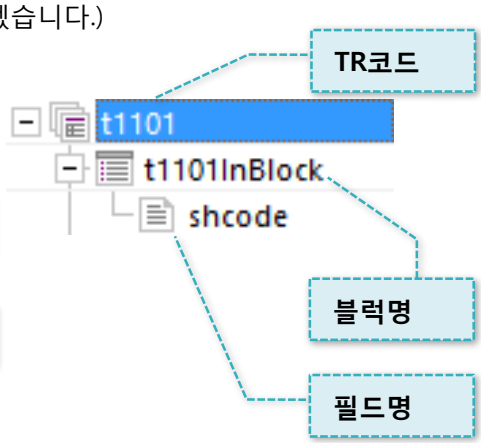
Step 4 입력데이터 설정

서버에 TR을 요청하기 이전에 InBlock 데이터를 입력해 주어야 합니다.
데이터를 입력하는 것은 XAQuery 객체의 SetFieldData 메쏘드를 사용하시면 됩니다.
t1101 res를 예로 들면 InBlock에는 종목코드라는 필드가 있습니다. 이 필드에 데이터를 넣은 후에 서버로 전송을 하시면 됩니다.

```
XAQuery_t1101.SetFieldData( "t1101InBlock", "shcode", 0, "078020" )
```

- 1번째 파라미터는 블럭명,
- 2번째 파라미터는 필드명,
- 3번째는 Occurs 일때의 인덱스 값이며, (여기서는 0을 넣도록 하겠습니다.)
- 4번째는 입력값입니다.





Step 5 입력데이터를 서버로 전송하기

서버로 TR을 전공하기 위해서는 XAQuery 객체의 Request 메쏘드를 사용하시면 됩니다.

```
XAQuery_t1101.Request( False )
```

- 첫번째 파라미터는 연속조회 여부입니다. 연속조회 일 경우에만 True이고 일반적으로는 False 입니다. (연속조회는 다음 장에서 설명합니다.)
- 리턴값은 정수이며 음수일 경우는 실패, 음수가 아닐 경우는 성공입니다.

Step 6 서버로부터 데이터 받기

입력데이터를 서버로 전송을 하였으면 서버로부터 데이터가 왔다는 이벤트를 받아야 합니다.
이벤트는 XAQuery 객체의 ReceiveData 이벤트를 사용하시면 됩니다.

```
XAQuery_t1101_ReceiveData( ByVal szTrCode As String )
```

이벤트를 받았다면 서버에서 수신된 데이터를 가져와야 합니다.
XAQuery 객체의 GetFieldData 메쏘드를 사용하면 데이터를 가져올 수 있습니다.

```
sName = XAQuery_t1101.GetFieldData( "t1101", "t1101OutBlock", "hname", 0 )
```

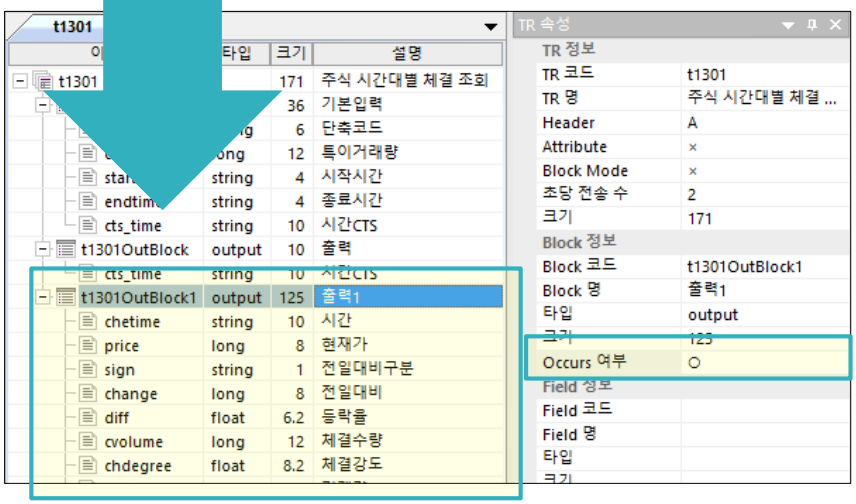
반복 데이터 조회 (Occurs)

Occurs는 프로그래밍 언어인 COBOL에서 나온 단어로 일련의 데이터가 반복해서 나오는 구조를 이야기 합니다.
API에서 Occurs 의 의미는 시간대별체결 화면의 그리드처럼 일련의 데이터가 반복해서 나올 경우를 정의합니다. 프로그래밍 언어에서의 배열(Array)와 같은 의미입니다.

※ Occurs 여부 확인방법

HTS 의 "[1301] 시간대별체결" 화면을 보시면 다음과 같이 시간대별 체결이 그리드에 표시가 됩니다.
"[1301] 시간대별체결" 화면은 t1301 TR을 사용해서 만들어진 부분으로 그리드 영역은 t1301OutBlock1에 해당합니다.
DevCenter의 TR Layout에서 t1301OutBlock1 을 선택하시면 TR속성창의 Occurs여부 속성에 Occurs 임을 표시합니다.

HTS의 [1301] 시간대별체결



DevCenter의 t1301 Layout

Step 1 Occurs 총 개수 알아오기

xingQ마스터의 "[1301] 시간대별체결" 화면의 그리드 총 행수는 10개입니다.

총 행수를 알아오는 방법은 XAQuery 객체의 GetBlockCount 메쏘드를 사용하시면 됩니다.

```
nCount = XAQuery_t1301.GetBlockCount( "t1301OutBlock1" )
```

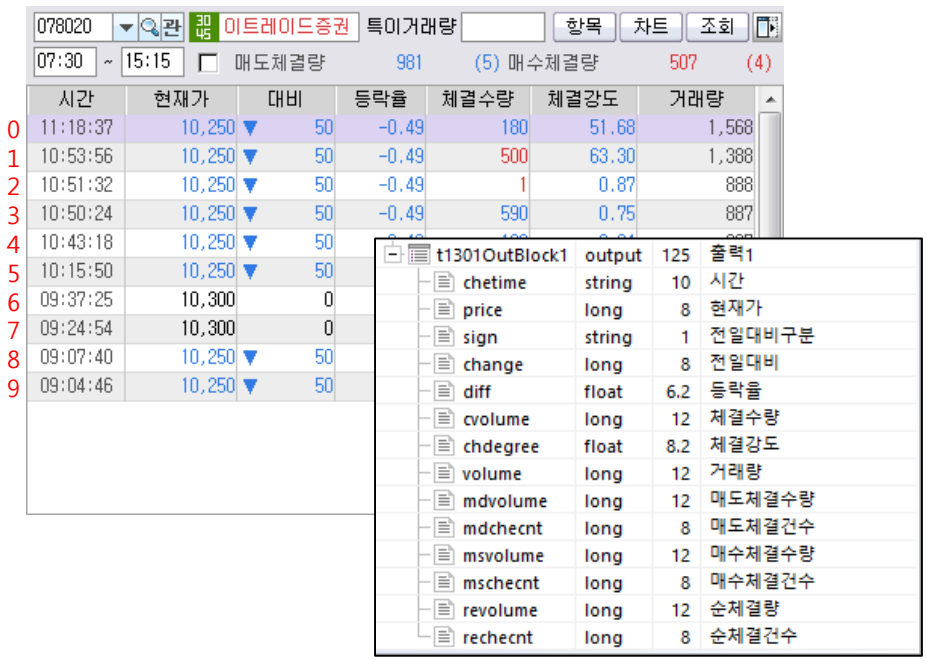
Step 2 Occurs의 데이터 가져오기

Occurs의 데이터 가져오는 방식은 일반과 같이 GetFieldData 메쏘드를 사용합니다.

"[1301] 시간대별체결" 화면의 행의 데이터를 가져오기 위해서는 3번째 파라미터에 행의 Index를 넣어 줘야 합니다.

```
Price = XAQuery_t1301.GetFieldData( "t1301OutBlock1", "price", Index )
```

Index는 1부터 시작하기 않고 오른쪽 그림처럼 0부터 시작하여 9까지의 Index를 넣을 수가 있습니다.



연속 데이터 조회

연속조회는 Occurs 데이터의 총 갯수가 너무 많아서 한번에 다 가져오지 못하는 것을 여러 번에 걸쳐서 가져오는 것입니다.

HTS(eBestPro)의 경우 스크롤바를 아래로 내렸을때 다시 요청해서 다음 데이터를 가져오거나 혹은 다음 버튼을 사용하여 다음 데이터를 가져오는 경우가 연속 조회에 해당됩니다.

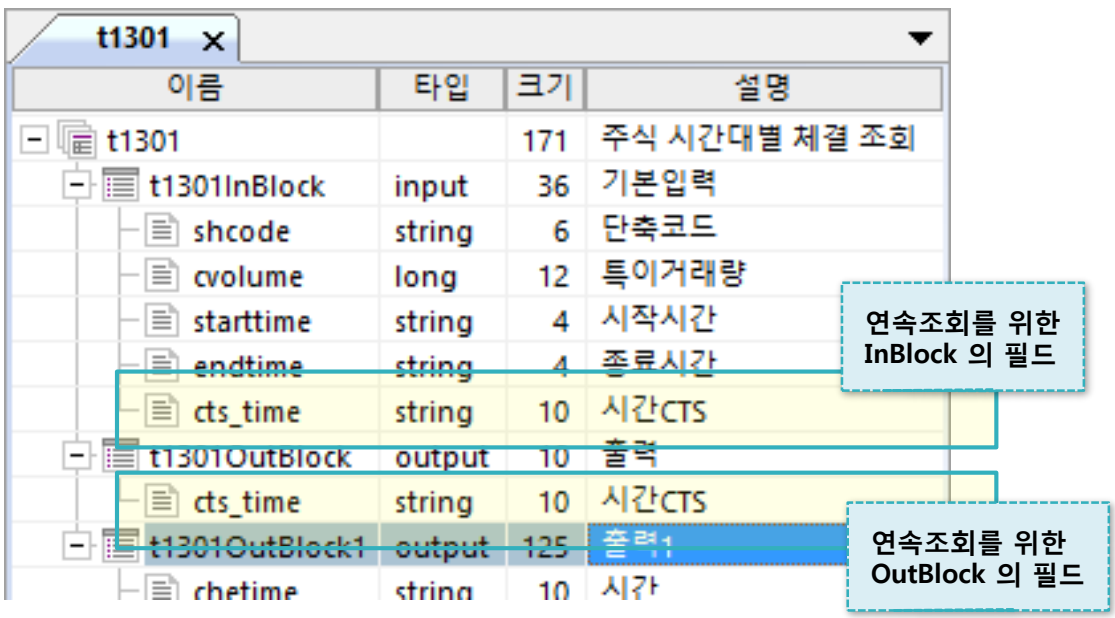
연속조회를 하기 위해서는
1. 다음에 가져올 데이터의 위치를 입력데이터에 넣어서
2. 서버에 연속조회 전송하기

를 해야 합니다.

1. 연속조회값 넣어주기

- 연속조회를 할 수 잇는 TR인 경우에는 TR의 Layout 을 보면 연속조회를 위한 필드가 존재합니다. (보통은 CTS_로 시작하는 필드명을 가지고 있습니다.)
- 입력값을 위해 InBlock에 연속조회 필드가 있고
- 다음값을 알려주기 위해 OutBlock에 연속조회 필드가 있습니다.
- OutBlock 의 연속조회용 필드의 값을 다음 조회시에 InBlock의 연속조회용 필드에 넣어주어야 합니다.

※ 계좌조회의 경우에는 이런 연속조회용 필드가 없을 수 있습니다. 이 경우에는 자동으로 연속조회값이 세팅됩니다.



Step 1 **연속조회 전송하기**

연속조회의 입력값을 입력하였다면 Request 메쏘를 이용하여 서버에 전송하여야 합니다. 이때, Request 의 파라미터를 TRUE로 하여 연속조회임을 서버에 알려줘야 합니다.

```
XAQuery_t1301_ReceiveData()
    ' OutBlock의 cts_time 을 가져와서
    ctsTime = XAQuery_t1301.GetFieldData( "t1301OutBlock", "cts_time", 0 )
    ' InBlock의 cts_time 에 넣고
    XAQuery_t1301.SetFieldData( "t1301InBlock", "cts_time", 0, ctsTime )
    ' 연속조회로 요청
    XAQuery_t1301.Request( True )
End Sub
```

실시간 데이터 조회

주식 현재가 같은 경우 장중에는 계속 바뀌게 됩니다.
조회TR의 경우 요청(Request)시점에서의 데이터를 전송해 주므로 그 이후 변경된 데이터를 받지 못합니다.
실시간 데이터 조회의 경우에는 (조회TR과는 다르게) 요청시점 이후에 데이터가 변경이 되면 그때 데이터를 전송해주며, 요청을 해제하기 전까지 데이터가 변경이 될 때마다 데이터를 전송해 줍니다.
다만, 요청시점 이후에 데이터가 변경이 되면 전송해 주기 때문에 요청시점에서의 데이터는 전송해주지 않습니다.
그러므로, 주식현재가를 구성하기 위해서는 조회TR로 요청시점의 데이터를 가져온 후에 실시간 데이터를 요청하여 그 이후 데이터를 실시간으로 받으셔야 합니다.

실시간 데이터는 XAReal 객체를 사용하며 XAQuery 와 사용방법이 대동소이 합니다.

Step 1 XAReal 선언 및 생성

```
' 객체선언 – 실시간TR코드는 무조건 3자리이다.
Dim WithEvents XAReal_S3_ As XAReal
' 객체생성
Set XAReal_S3_ = CreateObject( "XA_DataSet.XAReal" )
```

Step 2 RES 등록

```
XAReal_S3_.ResFileName = "S3_.res"
```

Step 3 데이터 입력

```
XAReal_S3_.SetFieldData( "InBlock", "shcode", 데이터 )
```

※ 계좌에 관한 실시간 정보일 경우에는 입력값이 없습니다.

Step 4 실시간 데이터 요청하기

```
' XAReal 객체의 AdviseRealData 메쏘드를 사용하여 데이터를 요청한다.
XAReal_S3_.AdviseRealData()
```

※ XAReal은 AdviseRealData()를 한 이후에 다른 종목도 실시간 시세를 받기를 원하시면
종목을 입력하신 후에 AdviseRealData()로 요청하시면 됩니다.

Step 5 실시간 데이터 받기

```
' XAReal 객체의 ReceiveRealData 이벤트로 데이터 받음
Sub XAReal_S3__ReceiveRealData( ByVal szTrCode )
    ' 현재가 가져오기
    XAReal_S3_.GetFieldData( "OutBlock", "price" )
End Sub
```

※ 실시간 데이터는 Occurs의 개념이 없이 단일 데이터입니다.

Step 6 실시간 데이터 요청 취소하기

더 이상 실시간 데이터가 필요 없을 경우 데이터 요청을 취소해야 합니다.
UnadviseRealData S3_에 걸려있는 모든 종목에 대해서 Unadvise 합니다.

```
XAReal_S3_.UnadviseRealData()
```

여러 종목에 대해서 한종목만 요청을 취소하시려면 UnadviseRealDataWithKey 메쏘드를 사용하시면 됩니다. 파라미터로 입력값을 넣으시면 됩니다. (예로, 당사 코드를 입력함)

```
XAReal_S3_.UnadviseRealDataWithKey( "078020" )
```

감사합니다