유안타증권
Yuanta Securities (Korea)

유안타 오픈 **API**

DLL API Reference

1 초기화

API 명	설 명
YOA_Initial	API 모듈을 초기화 합니다.
YOA_UnInitial	API 모듈이 사용하는 리소스를 해제합니다.

2 로그인

API 명	설 명
YOA_Login	API 서버에 로그인합니다.
YOA_Logout	API 서버에 로그아웃합니다.

3 조회 TR(DSO)

API 명	설 명
YOA_Request	조회 TR(DSO)를 API 서버에 요청합니다.
YOA_ReleaseData	수신 데이터와 조회 요청 정보를 삭제하고 Request ID 를 해제합니다.
YOA_Reset	조회 요청한 InBlock 과 결과 OutBlock 의 연결을 종료합니다. OnReceiveData 에서 동일 TR 요청 시 사용합니다.

4 실시간 TR(AUTO)

API 명	설 명
YOA_RegistAuto	실시간 TR(AUTO)을 API 서버에 등록합니다.
YOA_UnRegistAuto	API 서버에 등록된 실시간 TR(주어진 AUTO ID)을 해제합니다.

YOA_UnRegistAutoWithKey	API 서버에 등록된 실시간 TR(주어진 AUTO ID, 주어진 Key: 종목코드)을 해제합니다.
YOA_UnRegistAutoWithReqID	API 서버에 등록된 실시간 TR(주어진 Request ID)을 해제합니다.

5 TR 사용

API 명	설 명
YOA_SetTRFieldString	입력받은 TR 내의 블록의 필드값을 문자열로 설정합니다.
YOA_SetTRFieldLong	입력받은 TR 내의 블록의 필드값을 long 으로 설정합니다.
YOA_SetTRFieldLong64	입력받은 TR 내의 블록의 필드값을 INT64 로 설정합니다.
YOA_SetTRFieldDouble	입력받은 TR 내의 블록의 필드값을 double 로 설정합니다.
YOA_SetTRFieldFloat	입력받은 TR 내의 블록의 필드값을 float 으로 설정합니다.
YOA_SetTRFieldWord	입력받은 TR 내의 블록의 필드값을 WORD 로 설정합니다.
YOA_SetTRFieldByte	입력받은 TR 내의 블록의 필드값을 byte 로 설정합니다.
YOA_GetTRFieldAttr	입력받은 TR 내의 블록의 필드의 속성을 취득합니다.
YOA_GetTRFieldString	입력받은 TR 내의 블록의 필드값을 문자열로 취득합니다.
YOA_GetTRFieldLong	입력받은 TR 내의 블록의 필드값을 long 으로 취득합니다.
YOA_GetTRFieldLong64	입력받은 TR 내의 블록의 필드값을 INT64 로 취득합니다.
YOA_GetTRFieldDouble	입력받은 TR 내의 블록의 필드값을 double 로 취득합니다.
YOA_GetTRFieldFloat	입력받은 TR 내의 블록의 필드값을 float 으로 취득합니다.
YOA_GetTRFieldWord	입력받은 TR 내의 블록의 필드값을 WORD 로 취득합니다.
YOA_GetTRFieldByte	입력받은 TR 내의 블록의 필드값을 byte 로 취득합니다.
YOA_GetRowCount	입력받은 TR 내의 블록의 데이터 Row Count 를 반환합니다.
YOA_SetTRInfo	YOA_SetFieldX 와 YOA_GetFieldX 에서 사용될 TR, 블록 정보를

	설정합니다.
YOA_SetFieldString	SetTRInfo 로 설정된 블록의 필드값을 문자열로 설정합니다.
YOA_SetFieldLong	SetTRInfo 로 설정된 블록의 필드값을 long 으로 설정합니다.
YOA_SetFieldLong64	SetTRInfo 로 설정된 블록의 필드값을 INT64 로 설정합니다.
YOA_SetFieldDouble	SetTRInfo 로 설정된 블록의 필드값을 double 로 설정합니다.
YOA_SetFieldFloat	SetTRInfo 로 설정된 블록의 필드값을 float 으로 설정합니다.
YOA_SetFieldWord	SetTRInfo 로 설정된 블록의 필드값을 WORD 로 설정합니다.
YOA_SetFieldByte	SetTRInfo 로 설정된 블록의 필드값을 byte 로 설정합니다.
YOA_GetFieldAttr	SetTRInfo 로 설정된 블록의 필드의 속성을 취득합니다.
YOA_GetFieldString	SetTRInfo 로 설정된 블록의 필드값을 문자열로 취득합니다.
YOA_GetFieldLong	SetTRInfo 로 설정된 블록의 필드값을 long 으로 취득합니다.
YOA_GetFieldLong64	SetTRInfo 로 설정된 블록의 필드값을 INT64 로 취득합니다.
YOA_GetFieldDouble	SetTRInfo 로 설정된 블록의 필드값을 double 로 취득합니다.
YOA_GetFieldFloat	SetTRInfo 로 설정된 블록의 필드값을 float 으로 취득합니다.
YOA_GetFieldWord	SetTRInfo 로 설정된 블록의 필드값을 WORD 로 취득합니다.
YOA_GetFieldByte	SetTRInfo 로 설정된 블록의 필드값을 byte 로 취득합니다.

6 계좌

API 명	설 명
YOA_GetAccountCount	계좌의 개수를 반환합니다.
YOA_GetAccount	계좌번호를 반환합니다.
YOA_GetAccountInfo	계좌관련 정보를 반환합니다.

7 정보

API 명	설 명
YOA_GetLastError	마지막 오류 코드를 반환합니다.
YOA_GetErrorMessage	주어진 오류코드의 메시지를 취득합니다.
YOA_GetCodeInfo	주어진 종목 코드의 정보를 취득합니다.
YOA_SetTimeout	서버에 TR 요청 후, 대기하는 시간을 설정합니다.

▣ YOA_Initial

유안타증권 오픈 API 를 사용하기 위해 초기화를 하는 메소드입니다.

■ 메소드 원형

```
long YOA_Initial (
        LPCTSTR     szURL,
        HWND        hWnd,
        LPCTSTR     szPath,
        long        nStartMsgID
)
```

■ 매개변수

변 수	설 명
szURL	연결할 HTS 서버 URL - 국내 모의투자 접속 : “simul.tradar.api.com” - 국내 및 해외주식 운영 접속 : “real.tradar.api.com” - 해외선물옵션 모의투자 접속 : “simul.tradarglobal.api.com” - 해외선물옵션 운영 접속 : “real.tradarglobal.api.com”
hWnd	시스템 메시지를 받을 Windows Handle
szPath	유안타증권 오픈 API 모듈 Path
nStartMsgID	응답 메시지 시작 ID

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, -1(RESULT_FAIL)이 반환됩니다.

■ 부가설명

hWnd 는 System Message 를 받아서 처리할 윈도우의 핸들입니다.

szPath 를 기준으로 필요한 DLL 를 로드하거나, TR 명세서 로드합니다. 빈문자열을 입력하면 사용자 프로그램의 실행위치를 기준으로 동작합니다.

nStartMsgID 는 유안타증권 오픈 API 에서 보내는 메시지의 시작 값입니다. 주어진 값에 API 에서 정의한 값을 더해서 메시지로 보내게 됩니다.

cf) nSTartMsgID + CMD_RECEIVE_ERROR

- API 정의 값

메시지		설 명
CMD_YOA_FAIL	**-1**	API Call 실패
CMD_YOA_SUCCESS	**1**	API Call 성공
CMD_SYSTEM_MESSAGE	**2**	System Message
CMD_LOGIN	**3**	서버로부터 로그인 결과 받았을 때 발생
CMD_LOGOUT	**4**	서버로부터 로그아웃 결과 받았을 때 발생
CMD_RECEIVE_ERROR	**5**	서버로부터 오류를 받았을 때 발생
CMD_RECEIVE_DATA	**6**	Request 로 요청한 데이터를 받았을 때 발생
CMD_RECEIVE_REAL_DATA	**7**	RegistAuto 로 등록한 실시간을 받았을 때 발생

■ 예시

```
if ( RESULT_SUCCESS == YOA_Initial( _T("tradar.api.com"), GetSafeHwnd(), NULL, WM_USER) )
{
        AfxMessageBox( _T("유안타 Open API가 초기화되었습니다.") );
}
else
{
        AfxMessageBox( _T("유안타 Open API가 초기화에 실패하였습니다.") );
}
```

▣ YOA_UnInitial

유안타증권 오픈 API 사용을 끝내기 위해 API 모듈에서 사용한 리소스를 해제하는 메소드입니다.

■ 메소드 원형

```
long YOA_UnInitial ( )
```

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, -1(RESULT_FAIL)이 반환됩니다.

▣ YOA_IsConnect

유안타증권 HTS 서버와 연결을 확인하는 메소드입니다.

■ 메소드 원형

```
long YOA_IsConnec ( )
```

■ 반환값

연결 시, 1000(RESULT_SUCCESS)이 반환됩니다.

미연결 시, -1(RESULT_FAIL)이 반환됩니다.

▣ YOA_Login

유안타증권 HTS 서버에 접속을 하고 유안타증권 계정으로 로그인을 하는 메소드입니다.

■ 메소드 원형

```
long YOA_Login (
        HWND        hWnd,
        LPCTSTR     szUserID,
        LPCTSTR     szUserPWD,
        LPCTSTR     szCertPWD,
)
```

■ 매개변수

변 수	설 명
hWnd	로그인 결과 메시지를 받을 Windows Handle
szUserID	유안타증권 계정 ID
szUserPWD	유안타증권 계정 비밀번호
szCertPWD	공인인증서 비밀번호

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

hWnd 는 비동기적인 로그인 결과를 받아서 처리할 윈도우의 핸들입니다.

szUserID 길이는 4 자리보다 길어야합니다.

szUserPWD 길이는 4 자리보다 길어야합니다.

szUserPWD 길이는 8 자리보다 길어야합니다.

■ 예시

```
long nResult = YOA_Login( GetSafeHwnd(), strID, strPwd, strCertPwd );
if ( RESULT_SUCCESS == nResult )
{
        AfxMessageBox( _T("로그인 요청이 되었습니다.") );
}
else
{
        AfxMessageBox( _T("로그인 요청이 실패하였습니다.") );
        TCHAR msg[2048] = {0,};

        YOA_GetErrorMessage( nResult, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ YOA_Request

서버에 조회 TR(DSO)을 요청하는 메소드입니다.

■ 메소드 원형

```
long YOA_Request (
        HWND        hWnd,
        LPCTSTR     szDSOID,
        BOOL        bReleaseData,
        long        nNextReqID,
)
```

■ 매개변수

변 수	설 명
hWnd	조회 TR 결과 메시지를 받을 Windows Handle
szDSOID	조회 TR(DSO)의 ID
bReleaseData	조회 결과 반환 후, 요청 정보와 Request ID 를 자동 해제할지 여부
nNextReqID	다음 조회를 할 Request ID

■ 반환값

성공 시, 1000(ERROR_MAX_CODE) 보다 큰 Request ID 를 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

hWnd 는 비동기적인 조회 TR 결과를 받아서 처리할 윈도우의 핸들입니다. 조회 TR 에 대한 결과는 CMD_RECEIVE_DATA 메시지로 전달됩니다. 특정 요청에 대한 결과는 반환 받은 Request ID 로 구분할 수 있습니다. CMD_RECEIVE_DATA 메시지 전달 시에 WPARAM 으로 Request ID 를, LPARAM 으로 조회 TR(DSO)ID 를 전달합니다.

szDSOID 는 조회 TR(DSO : Data Service Object)의 ID 입니다.

bReleaseData 는 조회 TR(DSO) 결과를 전달한 후에, 조회를 위해 사용된 정보와 수신한 데이터의 메모리와 Request ID 를 해제를 자동으로 할 지 여부입니다. 기본적으로 자동 해지를 사용하며, 다음 조회와 같은 연속조회가 필요한 경우에만 FALSE 를 사용합니다.

nNextReqID 는 다음 조회를 할 조회 TR(DSO)의 처음 요청 시, 받은 Request ID 입니다. 다음 조회가 아닌 경우에는 -1(ERROR_MAX_CODE 보다 작은 수 : 1000)를 설정합니다.

연속조회의 경우는 이후 설명이 나오는 YOA_SetXXXX 메소드를 이용하여 필드 값들을 다시 설정할 필요가 없습니다.
최초 요청 시, 설정한 값을 보관하고 있다가 연속조회 YOA_Request 요청 시, 자동으로 필드 값이 설정됩니다.
(단, YOA_ReleaseData 로 해당 정보를 해제하지 않은 경우)

■ 예시

```
long nResult = YOA_Request( GetSafeHwnd(), _T("300001"), TRUE, -1 );
if ( ERROR_MAX_CODE < nResult )
{
        AfxMessageBox( _T("조회 요청이 되었습니다.") );
}
else
{
        AfxMessageBox( _T("조회 요청이 실패하였습니다.") );
        TCHAR msg[2048] = {0,};

        YOA_GetErrorMessage( nResult, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ YOA_ReleaseData

수신 데이터와 조회 요청 정보를 삭제하고 Request ID 를 해제하는 메소드입니다.

■ 메소드 원형

```
long YOA_ReleaseData (
        long            nReqID,
)
```

■ 매개변수

변 수	설 명
nReqID	해제할 Request ID

■ 반환값

성공 시, 1000(ERROR_MAX_CODE) 보다 큰 Request ID 를 반환됩니다.
실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

YOA_Request 에서 bReleaseData = FALSE 로 전달한 Request ID 만 정상 동작합니다. TRUE 인 경우 CMD_RELEASE_DATA 메시지 처리 후, 자동 해제되어 호출할 필요가 없습니다.
해제된 Request ID 를 다시 해제하는 경우 ERROR_REQUEST_NOT_FOUND 가 발생합니다.

bReleaseData = FALSE 로 전달하고 YOA_ReleaseData 를 호출하지 않으면 사용하지 않는 메모리가 쌓이게 되어 메모리가 부족한 현상이 발생할 수도 있습니다.

연속조회가 끝나지 않은 상태에서 YOA_ReleaseData 를 호출하게 되면 다음 연속조회가 동작하지 않습니다.

아직 결과를 수신하지 않은 Request ID 를 해제한 경우, 이후 결과를 수신 받을 수 없습니다.

■ 예시

```
long nResult = YOA_ReleaseData( 10001 );
if ( RESULT_SUCCESS != nResult )
{
        AfxMessageBox( _T("요청 데이터 및 Request ID 해제가 실패하였습니다.") );
        TCHAR msg[2048] = {0,};

        YOA_GetErrorMessage( nResult, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ YOA_Reset

조회 요청한 InBlock 과 결과 OuBlock 의 연결을 종료하는 메소드입니다.

■ 메소드 원형

```
long YOA_Reset (
        LPCTSTR      szDSOID
)
```

■ 매개변수

변 수	설 명
szDSOID	조회 TR(DSO)의 ID

■ 반환값

정상적으로 연결 종료 시, 1000(RESULT_SUCCESS)이 반환됩니다.
실패 시, -1(RESULT_FAIL)이 반환됩니다.

■ 부가설명

조회 요청의 결과를 수신하게 되면 WMU_RECEIVE_DATA 메시지를 수신하게 됩니다.
해당 메시지를 받으면 OutBlock 에 접근하여 결과 데이터를 사용할 수 있습니다.
이때 해당 OutBlock 은 요청한 데이터인 InBlock 과 연결되어 있습니다.
이는 결과 데이터 처리 시, 요청 했던 데이터도 가져와 사용할 수 있도록 지원하기 위함입니다.

이때 WMU_RECEIVE_DATA 처리 함수에서 바로 동일 TR 을 사용하기 위해 InBlock 에 입력 필드값을 설정하고 조회 요청을 하게 되면 이전 InBlock 의 데이터가 요청되어 동일 결과를 수신하게 됩니다. (다른 TR 사용은 상관 없습니다.)

이경우, InBlock 에 입력 필드값을 설정하기 전에 YOA_Reset("DSOID")를 호출하면 InBlock OutBlock 의 연결이 종료되어 정상적인 결과를 수신할 수 있습니다.

▣ YOA_RegistAuto

서버에 실시간 TR(AUTO)을 등록하는 메소드입니다.

■ 메소드 원형

```
long YOA_RegistAuto (
        HWND        hWnd,
        LPCTSTR     szAutoID
)
```

■ 매개변수

변 수	설 명
hWnd	실시간 TR 데이터 메시지를 받을 Windows Handle
szAutoID	실시간 TR(Auto)의 ID

■ 반환값

성공 시, 1000(ERROR_MAX_CODE) 보다 큰 Request ID 를 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

hWnd 는 비동기적인 실시간 TR 데이터를 받아서 처리할 윈도우의 핸들입니다. 실시간 TR 에 대한 데이터는 CMD_RECEIVE_REAL_DATA 메시지로 전달됩니다. 특정 요청에 대한 데이터는 반환 받은 Request ID 로 구분할 수 있습니다. CMD_RECEIVE_REAL_DATA 메시지 전달 시에 WPARAM 으로 Request ID 를, LPARAM 으로 실시간 TR(AUTO)ID 를 전달합니다.

szAutoID 는 실시간 TR(AUTO)의 ID 입니다.

■ 예시

```
long nResult = YOA_RegistAuto( GetSafeHwnd(), _T("11") );
```

▣ YOA_UnRegistAuto

서버에 등록된 실시간 TR(AUTO)중 특정 hWnd 에 연동된 실시간을 해제하는 메소드입니다.

■ 메소드 원형

```
long YOA_UnRegistAuto (
        HWND        hWnd,
        LPCTSTR     szAutoID
)
```

■ 매개변수

변 수	설 명
hWnd	실시간 TR 데이터 메시지를 받을 Windows Handle
szAutoID	실시간 TR(Auto)의 ID

■ 반환값

성공 시, 1000(ERROR_MAX_CODE) 보다 큰 Request ID 를 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

서버에 등록된 실시간(AUTO)을 해제하지 않으면, 화면에서 사용하지 않아도 실시간이 계속 내려와 프로그램 성능에 영향을 줄 수 있습니다.

■ 예시

```
if ( RESULT_FAIL == YOA_UnRegistAuto( GetSafeHwnd(), _T("11") ) )
{
        TCHAR msg[2048] = {0,};
        YOA_GetErrorMessage( nResult, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ 유사 지원 메서드

■ YOA_UnRegistWithKey(

HWND hWnd,
LPCTSTR szAutoID,
LPCTSTR szKey

)

- 서버에 등록된 실시간 TR(AUTO)중 주어진 hWnd 에 연동된 특정 키의 실시간을 해제하는 메소드입니다. 키는 실시간 TR(AUTO)마다 다르며, 보통 종목 코드입니다.

■ YOA_UnRegistWithReqID(

long nReqID

)

- 서버에 등록된 실시간 TR(AUTO)중 특정 Requst ID 실시간을 해제하는 메소드 입니다. Request ID 는 YOA_RegistAuto 호출 시, 반환됩니다.

■ YOA_UnRegistWithWindow(

HWND hWnd

)

- 서버에 등록된 실시간 TR(AUTO)중 특정 hWnd 에 연동된 모든 실시간을 해제하는 메소드입니다.

▣ YOA_SetTRFieldString

TR(DSO, AUTO) 블록의 필드 데이터를 문자열로 설정하는 메소드입니다.

■ 메소드 원형

```
long YOA_SetTRFieldString (
        LPCTSTR     szTRID,
        LPCTSTR     szBlockName,
        LPCTSTR     szFieldName,
        LPCTSTR     szValue,
        long        nOccursIndex
)
```

■ 매개변수

변 수	설 명
szTRID	필드를 설정할 TR(조회:DSO, 실시간:AUTO)의 ID
szBlockName	TR 의 블록명
szFieldName	블록의 필드명
szValue	설정 값
nOccursIndex	블록이 배열인 경우, 설정 값이 위치할 Index

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

szValue 는 설정할 필드의 값입니다. 필드의 데이터 타입이 문자열이 아니어도 해당 데이터 타입으로 변환하여 설정합니다.

nOccursIndex 는 Block 인 배열(Occurs) 타입인 경우, 설정하는 데이터가 위치할 Index 입니다. 기본적으로는 0 을 사용합니다.

■ 예시

```
long result = YOA_SetTRFieldString( _T("300001"), _T("InBlock1"), _T("jongcode"),
_T("003470"), 0 );

if ( RESULT_SUCCESS != result )
{
        TCHAR msg[1024] = {0,};

        YOA_GetErrorMessage( result, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ 관련 지원 메서드

■ YOA_SetTRFieldLong

- TR(DSO, AUTO) 블록의 필드 데이터를 long 값으로 설정하는 메소드입니다.

■ YOA_SetTRFieldDouble

- TR(DSO, AUTO) 블록의 필드 데이터를 double 값으로 설정하는 메소드입니다.

■ YOA_SetTRFieldFloat

- TR(DSO, AUTO) 블록의 필드 데이터를 float 값으로 설정하는 메소드입니다.

■ YOA_SetTRFieldWord

- TR(DSO, AUTO) 블록의 필드 데이터를 WORD 값으로 설정하는 메소드입니다.

■ YOA_SetTRFieldByte

- TR(DSO, AUTO) 블록의 필드 데이터를 byte 값으로 설정하는 메소드입니다.

※ 설명은 YOA_SetTRFieldString 을 참고하시기 바랍니다.

▣ YOA_SetTRInfo

YOA_SetTRFieldXXX 에서 사용할 TR(DSO, AUTO)명, 블록명을 설정하는 메소드입니다.

■ 메소드 원형

```
long YOA_SetTRInfo (
        LPCTSTR     szTRID,
        LPCTSTR     szBlockName
)
```

■ 매개변수

변 수	설 명
szTRID	필드를 설정할 TR(조회:DSO, 실시간:AUTO)의 ID
szBlockName	TR 의 블록명

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

YOA_SetTRFieldXXX 는 TR 명과 Block 명이 필요한 메소드이기 때문에 많은 필드를 사용할 경우 TR 명과 Block 명을 반복적으로 코딩해야 하지만 YOA_SetFieldXXX 는 YOA_SetTRInfo 로 한번 설정하여 사용합니다.

■ 예시

```
YOA_SetTRInfo( _T("300001"), _T("InBlock1") );
YOA_SetFieldLong( _T("jang"), 1, 0 );
YOA_SetFieldString( _T("jongcode"), _T("003470"), 0 );
YOA_SetFieldString( _T("outflag"), _T("N"), 0 );
```

▣ YOA_SetFieldString

YOA_SetTRInfo 로 설정한 TR(DSO, AUTO) 블록의 필드 데이터를 문자열로 설정하는 메소드입니다.

■ 메소드 원형

```
long YOA_SetTRFieldString (
        LPCTSTR     szFieldName,
        LPCTSTR     szValue,
        long        nOccursIndex
)
```

■ 매개변수

변 수	설 명
szFieldName	블록의 필드명
szValue	설정 값
nOccursIndex	블록이 배열인 경우, 설정 값이 위치할 Index

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

szValue 는 설정할 필드의 값입니다. 필드의 데이터 타입이 문자열이 아니어도 해당 데이터 타입으로 변환하여 설정합니다.

nOccursIndex 는 Block 인 배열(Occurs) 타입인 경우, 설정하는 데이터가 위치할 Index 입니다. 기본적으로는 0 을 사용합니다.

이 메소드를 사용하기 전에 반드시 YOA_SetTRInfo 로 TR 명과 블록명을 설정해야 합니다. 설정하지 않은 경우, 이전에 YOA_SetTRInfo 로 설정한 TR 명과 블록명이 적용되어 블록을 찾을 수 없는 오류가 발생하거나, 필드를 찾을 수 없는 오류가 발생할 수 있습니다.

■ 예시

```
YOA_SetTRInfo( _T("300001"), _T("InBlock1") );
long result = YOA_SetFieldString( _T("jongcode"), _T("003470"), 0 );
if ( RESULT_SUCCESS != result )
{
        TCHAR msg[1024] = {0,};

        YOA_GetErrorMessage( result, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ 관련 지원 메서드

■ YOA_SetFieldLong

- TR(DSO, AUTO) 블록의 필드 데이터를 long 값으로 설정하는 메소드입니다.

■ YOA_SetFieldDouble

- TR(DSO, AUTO) 블록의 필드 데이터를 double 값으로 설정하는 메소드입니다.

■ YOA_SetFieldFloat

- TR(DSO, AUTO) 블록의 필드 데이터를 float 값으로 설정하는 메소드입니다.

■ YOA_SetFieldWord

- TR(DSO, AUTO) 블록의 필드 데이터를 WORD 값으로 설정하는 메소드입니다.

■ YOA_SetFieldByte

- TR(DSO, AUTO) 블록의 필드 데이터를 byte 값으로 설정하는 메소드입니다.

※ 설명은 YOA_SetFieldString 을 참고하시기 바랍니다.

▣ YOA_GetTRFieldString

TR(DSO, AUTO) 블록의 필드 데이터를 문자열로 취득하는 메소드입니다.

■ 메소드 원형

```
long YOA_SetTRFieldString (
        LPCTSTR     szTRID,
        LPCTSTR     szBlockName,
        LPCTSTR     szFieldName,
        LPCTSTR     szValueBuf,
        long        nValueBufSize,
        long        nOccursIndex
)
```

■ 매개변수

변 수	설 명
szTRID	필드를 설정할 TR(조회:DSO, 실시간:AUTO)의 ID
szBlockName	TR 의 블록명
szFieldName	블록의 필드명
szValueBuf	필드 값을 받을 문자열 버퍼
nValueBufSize	문자열 버퍼의 크기
nOccursIndex	블록이 배열인 경우, 설정 값이 위치할 Index

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

szValueBuf 는 필드 값을 받을 문자열 버퍼입니다. 필드의 데이터 타입이 문자열이 아니어도 문자열로 변환하여 취득할 수 있습니다.

nValueBufSize 는 필드 값을 받을 문자열 버퍼의 크기로 필드 값보다 작은 버퍼인 경우는 필드 값이 버퍼의 사이즈만큼만 취득됩니다.

nOccursIndex 는 Block 인 배열(Occurs) 타입인 경우, 설정하는 데이터가 위치할 Index 입니다. 기본적으로는 0 을 사용합니다.

■ 예시

```
TCHAR data[1024] = {0,};
memset( data, 0x00, sizeof(data) );

long result = YOA_GetTRFieldString( _T("300001"), _T("OutBlock2"), _T("highjuka"), data,
sizeof(data), 1 );

if ( RESULT_SUCCESS != result )
{
        TCHAR msg[1024] = {0,};

        YOA_GetErrorMessage( result, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ 관련 지원 메서드

■ YOA_GetTRFieldAttr

- TR(DSO, AUTO) 블록의 필드 데이터의 속성값(색상 구분값)을 취득하는 메소드입니다.

■ YOA_GetTRFieldLong

- TR(DSO, AUTO) 블록의 필드 데이터를 long 타입으로 취득하는 메소드입니다.

■ YOA_GetTRFieldDouble

- TR(DSO, AUTO) 블록의 필드 데이터를 double 값으로 설정하는 메소드입니다.

■ YOA_GetTRFieldFloat

- TR(DSO, AUTO) 블록의 필드 데이터를 float 값으로 설정하는 메소드입니다.

■ YOA_GetTRFieldWord

- TR(DSO, AUTO) 블록의 필드 데이터를 WORD 값으로 설정하는 메소드입니다.

■ YOA_GetTRFieldByte

- TR(DSO, AUTO) 블록의 필드 데이터를 byte 값으로 설정하는 메소드입니다.

※ 설명은 YOA_GetTRFieldString 을 참고하시기 바랍니다.

▣ YOA_GetFieldString

YOA_SetTRInfo 로 설정한 TR(DSO, AUTO) 블록의 필드 데이터를 문자열로 취득하는 메소드입니다.

■ 메소드 원형

```
long YOA_GetFieldString (
        LPCTSTR     szFieldName,
        LPCTSTR     szValueBuf,
        long        nValueBufSize,
        long        nOccursIndex
)
```

■ 매개변수

변 수	설 명
szFieldName	블록의 필드명
szValueBuf	필드 값을 받을 문자열 버퍼
nValueBufSize	문자열 버퍼의 크기
nOccursIndex	블록이 배열인 경우, 설정 값이 위치할 Index

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

szValueBuf 는 필드 값을 받을 문자열 버퍼입니다. 필드의 데이터 타입이 문자열이 아니어도 문자열로 변환하여 취득할 수 있습니다.

nValueBufSize 는 필드 값을 받을 문자열 버퍼의 크기로 필드 값보다 작은 버퍼인 경우는 필드 값이 버퍼의 사이즈만큼만 취득됩니다.

nOccursIndex 는 Block 인 배열(Occurs) 타입인 경우, 설정하는 데이터가 위치할 Index 입니다. 기본적으로는 0 을 사용합니다.

■ 예시

```
TCHAR data[1024] = {0,};
memset( data, 0x00, sizeof(data) );

YOA_SetTRInfo( _T("300001"), _T("OutBlock2") );
long result = YOA_GetFieldString( _T("highjuka"), data, sizeof(data), 1 );

if ( RESULT_SUCCESS != result )
{
        TCHAR msg[1024] = {0,};

        YOA_GetErrorMessage( result, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ 관련 지원 메서드

■ YOA_GetFieldAttr

- TR(DSO, AUTO) 블록의 필드 데이터의 속성값(색상 구분값)을 취득하는 메소드입니다.

■ YOA_GetFieldLong

- TR(DSO, AUTO) 블록의 필드 데이터를 long 타입으로 취득하는 메소드입니다.

■ YOA_GetFieldDouble

- TR(DSO, AUTO) 블록의 필드 데이터를 double 값으로 설정하는 메소드입니다.

■ YOA_GetFieldFloat

- TR(DSO, AUTO) 블록의 필드 데이터를 float 값으로 설정하는 메소드입니다.

■ YOA_GetFieldWord

- TR(DSO, AUTO) 블록의 필드 데이터를 WORD 값으로 설정하는 메소드입니다.

■ YOA_GetFieldByte

- TR(DSO, AUTO) 블록의 필드 데이터를 byte 값으로 설정하는 메소드입니다.

※ 설명은 YOA_GetFieldString 을 참고하시기 바랍니다.

▣ YOA_GetRowCount

TR(DSO, AUTO) 블록이 Occurs 인 경우, 해당 블록의 Row 수를 반환합니다.

■ 메소드 원형

```
long YOA_GetRowCount (
        LPCTSTR     szTRID,
        LPCTSTR     szBlockName
)
```

■ 매개변수

변 수	설 명
szTRID	TR(조회:DSO, 실시간:AUTO)의 ID
szBlockName	TR 의 블록명

■ 반환값

성공 시, Row 수가 반환됩니다.

실패 시, 해당 -1(RESULT_FAIL)을 반환합니다. YOA_GetLastError 와 YOA_GetErrorMessage 메소드를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 예시

```
long count = YOA_GetRowCount( _T("300001"), _T("OutBlock2") );

if ( RESULT_FAIL == count )
{
        Int error_code = YOA_GetLastError();
        TCHAR msg[1024] = {0,};

        YOA_GetErrorMessage(error_code, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ YOA_GetAccountCount

계좌의 개수를 반환합니다.

■ 메소드 원형

```
long YOA_GetAccountCount ( )
```

■ 반환값

성공 시, Row 수가 반환됩니다.

실패 시, 해당 -1(RESULT_FAIL)을 반환합니다. YOA_GetLastError 와 YOA_GetErrorMessage 메소드를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 예시

```
long count = YOA_GetRowCount( _T("300001"), _T("OutBlock2") );

if ( RESULT_FAIL == count )
{
        Int error_code = YOA_GetLastError();
        TCHAR msg[1024] = {0,};

        YOA_GetErrorMessage(error_code, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ YOA_GetAccount

계좌번호를 취득하는 메소드입니다.

■ 메소드 원형

```
long YOA_GetAccount (
        long        nIndex,
        LPCTSTR     szAccounBuf,
        long        nAccountBufSize
)
```

■ 매개변수

변 수	설 명
nIndex	취득한 계좌번호의 Index
szAccountBuf	계좌번호를 받을 문자열 버퍼
nAccountBufSize	문자열 버퍼의 크기

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

nIndex 는 YOA_GetAccountCount 의 반환 값보다 작아야만 합니다. 음수 값이나 계좌목록 수보다 큰 Index 로 요청하는 경우 ERROR_INDEX_OUT_OF_BOUNDS 가 발생합니다.

계좌번호는 12 자리입니다. 문자열 버퍼는 최소 12 바이트 이상을 할당해야 합니다.
12 바이트보다 작은 경우, 계좌번호의 일부만 취득 됩니다.

■ 예시

```
TCHAR account[64];

long result = YOA_GetAccount( 0, account, sizeof(account) );
if ( RESULT_FAIL == result)
{
        TCHAR msg[1024] = {0,};

        YOA_GetErrorMessage(result, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ YOA_GetAccountInfo

계좌 정보를 취득하는 메소드입니다.

■ 메소드 원형

```
long YOA_GetAccountInfo (
        long        nInfoType,
        LPCTSTR     szAccount,
        LPCTSTR     szAcctInfoBuf
        long        nAcctInfoBufSize
)
```

■ 매개변수

변 수	설 명
nInfoType	계좌 정보 구분 값
szAccount	계좌번호
szAcctInfoBuf	계좌 정보를 받을 문자열 버퍼
nAcctInfoBufSize	문자열 버퍼의 크기

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, 해당 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

■ 부가설명

nInfoType 은 취득하고자 하는 계좌 정보의 구분 값입니다.

* 1(ACCOUNT_INFO_NAME) : 계좌명
* 2(ACCOUNT_INFO_NICKNAME) : 계좌별명
* 3(ACCOUNT_INFO_TYPE) : 상품구분

■ 예시

```
TCHAR acctInfo[1024];

long result = YOA_GetAccountInfo( 1, _T("0000-0000-0000"), acctInf, sizeof(acctInfo) );
if ( RESULT_FAIL == result)
{
        TCHAR msg[1024] = {0,};

        YOA_GetErrorMessage(result, msg, sizeof(msg) );
        AfxMessageBox( msg );
}
```

▣ YOA_GetLastError

마지막 오류 코드를 반환하는 메소드입니다.

■ 메소드 원형

```
long YOA_GetLastError ( )
```

■ 반환값

마지막 오류코드를 반환합니다. YOA_GetErrorMessage 를 통해 오류 내용을 문자열로 확인할 수 있습니다.

▣ YOA_GetErrorMessage

주어진 오류 코드의 내용을 문자열로 취득하는 메소드입니다.

■ 메소드 원형

```
long YOA_GetErrorMessage (
        long        nErrorCode
        LPCTSTR     szErrorMsgBuf
        long        nErrorMsgBufSize
)
```

■ 매개변수

변 수	설 명
nErrorCode	에러코드
szErrorMsgBuf	오류 내용을 받을 문자열 버퍼
nErrorMsgBufSize	문자열 버퍼의 크기

■ 반환값

성공 시, 오류 내용 문자열의 길이가 반환됩니다.

실패 시, 0 이 반환됩니다.

■ 예시

```
TCHAR msg[1024] = {0,};

YOA_GetErrorMessage( 11, msg, sizeof(msg) );
AfxMessageBox( msg );
```

▣ YOA_GetCodeInfo

종목 정보를 취득하는 메소드입니다.

■ 메소드 원형

```
long YOA_GetAccountInfo (
        long        nMarketType,
        long        nInfoType,
        LPCTSTR     szCode,
        LPCTSTR     szCodeInfoBuf
        long        nCodeInfoBufSize
)
```

■ 매개변수

변 수	설 명
nMarketType	종목 시장 구분 값
nInfoType	종목 정보 구분 값
szCode	종목 코드
szCodeInfoBuf	종목 정보를 받을 문자열 버퍼
nCodeInfoBufSize	문자열 버퍼의 크기

■ 반환값

성공 시, 1000(RESULT_SUCCESS)이 반환됩니다.

실패 시, -1(RESULT_FAIL)이 반환됩니다.

■ 부가설명

nMarketType 은 취득하고자 하는 종목의 시장 구분 값 입니다.

* 0(MARKET_TYPE_INTERNAL) : 국내시장(국내주식, 국내선물옵션)

* 1(MARKET_TYPE_GLOBAL_STOCK) : 해외주식

* 2(MARKET_TYPE_GLOBAL_DERIVATIVE) : 해외선물옵션(향후 지원 예정)

nInfoType 은 취득하고자 하는 종목 정보의 구분 값입니다.

* 1(CODE_INFO_NAME) : 한글 종목명
* 2(CODE_INFO_ENG_NAME) : 영문 종목명
* 3(CODE_INFO_JANG_GUBUN) : 장구분

■ 예시

```
TCHAR codeInfo[64];

YOA_GetCodeInfo( 0, 3, codeInf, sizeof(codeInfo) );
```