

Spreadsheet Server Administration Manual

Version 3.2.1

Copyright Information

This document is protected by copyright law and may not be reproduced or distributed either in part or in total.

The licensee is not allowed to pass on the software or the accompanying written materials to third parties or make them otherwise available without prior written agreement of the licensor.

Information in this manual that refers to possible product extensions or to available accessories is not legally binding, especially because the products are subject to continuous adaptation and because the information may also relate to future development. The contents of this manual can change without prior notice and does not represent any legal obligation on the part of Swissrisk AG.

Swissrisk AG cannot be made liable for the correctness of information in this manual nor for damages resulting from the use of this information or the impossibility of using this information.

All other legal regulations for using the software and the corresponding documentation are set in the applicable license agreement.

Spreadsheet Server, InVision, Slingshot 2, +Arena and IAW are trademarks of Swissrisk AG.

All other product and company names mentioned in this manual are trademarks of their respective companies.

You are welcome to let us know how to improve this documentation. Please mail to documentation@swissrisk.com.

Version May 2008

Contents

Introduction.....	6
1 Restrictions and Limitations.....	7
2 Configuration	8
Cache Size	8
Time	8
Miscellaneous.....	9
Contribution.....	9
Bid/Ask grouping	10
Permissioning	10
Cache File	11
Chain of all objects	11
Command line argument	12
Environment variable adjustments	12
3 Publishing from a spreadsheet	14
Publisher Functions.....	14
Publish Rate (PR) Function.....	14
Publish Rate Extended (PREx) Function	15
Publish Rate Single (PRE1) Function	15
Refresh Updates (REFP) Function	15
Drop Record (DROPP) Function.....	16
Clear Record (CLEARP) Function	16
Error Handling	16
Using the functions.....	17
4 Contributing from a Spreadsheet.....	18
Contribution Functions	18
Contribute Rate (CR) Function.....	18
Contribute Rate Extended (CREx) Function	18
Contribute Rate Single (CRE1) Function	19
Refresh Contributions (REFC) Function	19
Error Handling	19
Using the functions.....	19
5 Example 1	21
6 Example 2.....	23
7 Example 3 - The Chains Sheet.....	25
Index	29

How to Use this Manual

These notes are provided to assist you in locating the information that you require. Please read this section before using the manual.

Where applicable, this manual uses the terminology of the Windows operating system, which it is assumed that you are familiar with.

In order to make it easier for you to find the required information, the following conventions are used in the manual:

Frame and bold type	For keys <u>Example:</u> Press Enter
Frame and bold type	For keyboard shortcuts <u>Example:</u> Press Ctrl+V
Bold type	The names of menus, menu options, commands and buttons appear in bold type. <u>Example:</u> The File menu
<i>Italics</i>	Field names as well as directories in the tree view are highlighted in italics. <u>Example:</u> <i>Stocks, Warrants, QRP</i>

Abbreviations

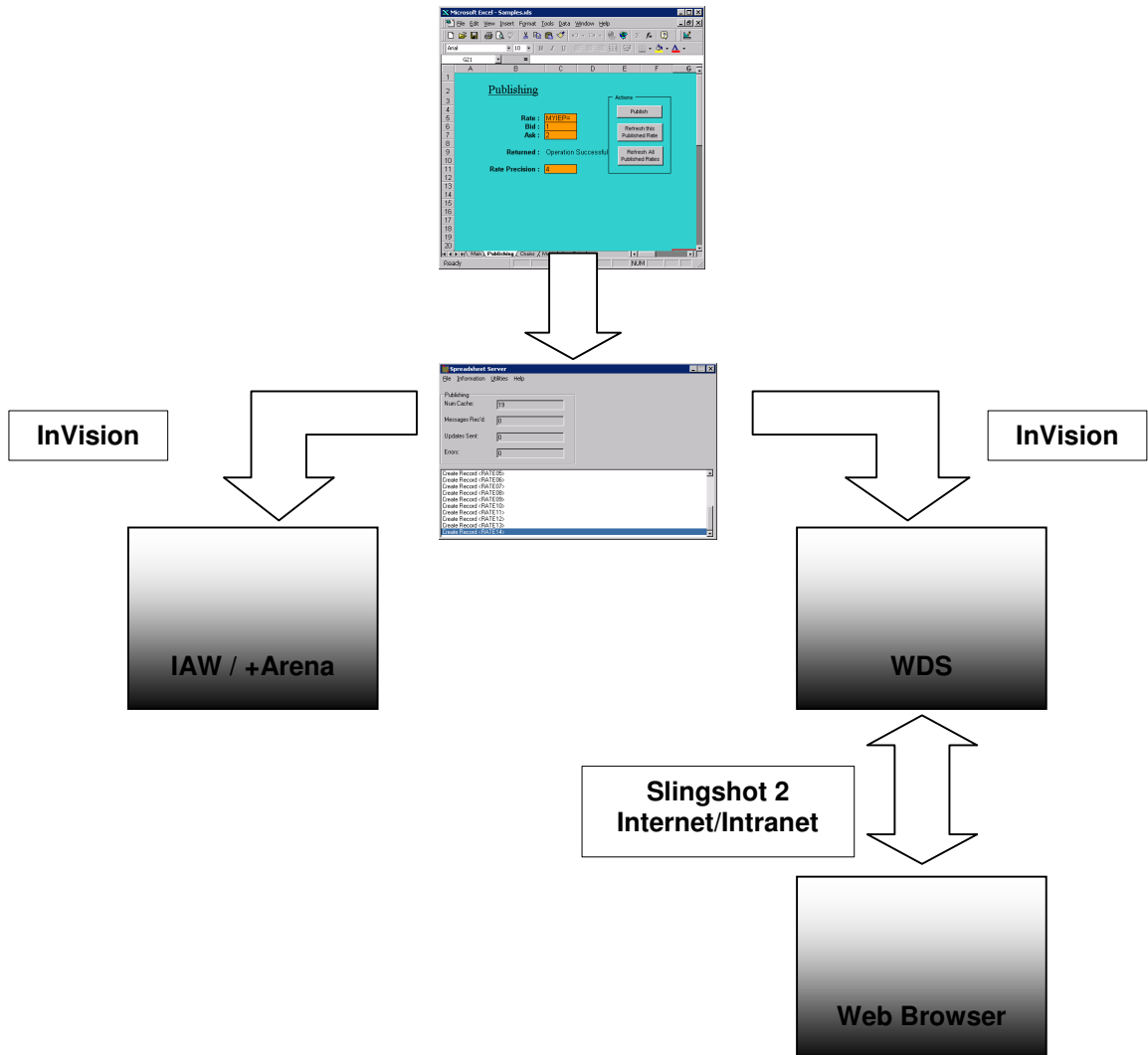
IAW	InVision Advanced Workstation
PE	Permission number
RIC	Reuters Instrument Code

Introduction

The Spreadsheet Server allows records to be published over InVision/Slingshot 2 distributions system. Records are entered through the use of Excel spreadsheets and the data is placed on the relevant service through the use of Windows messaging and the Spreadsheet Server application.

As the Spreadsheet Server application communicates with the InVision/Slingshot 2 distribution systems, the Spreadsheet Server application may act as an independent publisher, to the Browser Java client and InVision workstations such as IAW and +Arena. These clients may connect to the Spreadsheet Server as it would to any supported financial server/feed.

The flow of data between the Excel Spreadsheets, the Spreadsheet Server application, and the desktop applications may be viewed as follows:



1 Restrictions and Limitations

The following are not configurable:

- Maximum record name length 63 characters
- Maximum contributor name length 15 characters
- Maximum length of data sent using **PR** function 4000 characters
- Maximum cacheable objects: limited by available memory
- Maximum number of services: 32

2 Configuration

Cache Size

Now basically unlimited – limited only by available memory and CPU power.

Time

The user can configure the Spreadsheet Server to automatically send the time with every update. This means that the user, i.e. spreadsheet does not have to send any time information with the updates. In order to send the time, the following parameters must be configured in the invision.ini file.

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER:	<i>SEND_TIME</i>
VALUE(S):	TRUE/FALSE
DEFAULT	TRUE
SYNTAX:	SEND_TIME = TRUE

This parameter determines whether the Spreadsheet Server tags on a time field to all updates sent out (default: yes).

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER:	<i>TIME_FID</i>
VALUE(S):	String
DEFAULT	5
SYNTAX:	TIME_FID = 5

This parameter determines the field ID in which the time information is sent.

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER:	<i>TIME_FORMAT</i>
VALUE(S):	HMS or HM
DEFAULT	HMS
SYNTAX:	TIME_FORMAT = HMS

This parameter determines the format of the time field value. There are two formats available namely HMS (hh:mm:ss) and HM (hh:mm).

Note that once this functionality is turned on, the Spreadsheet will send the time with all updates as long as there is no value sent by the user for the time field.

Miscellaneous

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	<i>MASK</i>
VALUE(S):	Integer
DEFAULT	3
SYNTAX:	MASK = 3

This parameter defines the mask number, which is assigned to records when they are created in the Publisher cache. The mask number is for display applications like +Arena or IAW to select the right display template when doing the layout for the display of the record requested. The mask number is not used in calculation systems like I-Pricer or I-Trader.

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	<i>DELIMITER</i>
VALUE(S):	single character
DEFAULT	; (semicolon)
SYNTAX:	DELIMITER = /

This parameter defines the delimiter that is used to separate the data and field IDs in the Publishing and Contribution functions. Note that this delimiter must not be part of field IDs or field values. Depending on the “regional settings” of Windows, numbers may contain a point and/or a comma – which cannot be used as delimiter.

Contribution

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	<i>CONTRIBUTION</i>
VALUE(S):	Boolean
DEFAULT	FALSE
SYNTAX:	CONTRIBUTION = TRUE

Switches on or off contribution from the Spreadsheet Server at all (as an InVision client to an InVision server. Default: off). This does not affect publishing data (as an InVision server).

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	<i>CONTRIB_USERNAME</i>
VALUE(S):	string
DEFAULT	GUEST
SYNTAX:	CONTRIB_USERNAME = SMITH

User name to be send with every contribution in field CONTRIB_USERNAME_FID (see below). If specified, but empty, no user name will be sent.

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	CONTRIB_USERNAME_FID
VALUE(S):	string
DEFAULT	3
SYNTAX:	CONTRIB_USERNAME_FID = 3

Field in which the CONTRIB_USERNAME (see above) will be send with every contribution. If specified, but empty, no user name will be sent.

Bid/Ask grouping

Sometimes it is a requirement that bid price and ask price are always sent together in an update, even if one of it has not changed. The Spreadsheet Server knows about this and will recognize “lonely” bid or ask fields and add the last known value for the other field.

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	BID_FID
VALUE(S):	string
DEFAULT	22
SYNTAX:	BID_FID = 22

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	ASK_FID
VALUE(S):	string
DEFAULT	25
SYNTAX:	ASK_FID = 25

If the Spreadsheet Server finds one of the bid and ask field specified here in an update to be published/contributed, it makes sure that the other field will also be included within this update (using the last known value, if there is no new value).

Permissioning

Objects in an InVision server (like the Spreadsheet Server) may have a permissioning information attached to it. This is a number (called “PE-number”, default: 0 = no access restrictions) which can be used to restrict access to this object. This PE number has to be set when the object is created (first time published from the spreadsheet or read from the cache file) and will not change later on.

(Another thing is “dynamic permissioning” – see the InVision documentation about this.)

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	PERM_FID
VALUE(S):	string
DEFAULT	0
SYNTAX:	PERM_FID = 4711

If PERM_FID is not empty and not "0", the initial update for an object (which will be used to create it) is scanned for the field ID specified here (e.g. "4711"). If found, the value of this field is converted to a number and used as PE number for the object.

Cache File

The Spreadsheet Server can store all published objects in a "cache file". If terminated and started again later, the Spreadsheet Server will read all objects from this cache file and publish them with the values they had when the cache file was written (in the last run of the Spreadsheet Server, may be hours/days ago). As these values may be out of date, they are marked as "stale". Once they get an update from the Spreadsheet (**PR()** function), their status is changed to "live" again, because now the data is definitively up-to-date.

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	<i>PUBLISHER_CACHE_FILE</i>
VALUE(S):	string
DEFAULT	sss_p.dat
SYNTAX:	PUBLISHER_CACHE_FILE = sss_cache.dat

Specify the name of the cache file. If an empty name is specified (parameter is present in **invision.ini** file, but with no value), no cache file is used at all.

For backward compatibility, the parameter PUBLISHER_CACHE_FILE is also checked, if PUBLISHER_CACHE_FILE is not specified.

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	<i>PRELOAD_STALE</i>
VALUE(S):	Boolean
DEFAULT	TRUE
SYNTAX:	PRELOAD_STALE = FALSE

By default, values read from the cache file are published with status "stale" to mark them as possibly not up to date, before they get an update from the spreadsheet.

If you want the data to be "live" (not stale) even if only read from the cache file, set PRELOAD_STALE to FALSE.

Chain of all objects

The Spreadsheet Server can create a Reuters-like chain containing all published objects. This provides an easy way to open all published objects in a frontend application like +Arena or IAW.

Depending on the number of published objects, the chain may have multiple records, their names are "<element number>#<base name>", e.g. "0#SHEETSERVER", "1#SHEETSERVER", ...

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	CHAIN_BASENAME
VALUE(S):	string
DEFAULT	SHEETSERVER
SYNTAX:	CHAIN_BASENAME = ALL_MY_OBJECTS

Base name of the chain, the part of the name after "#".

FILE:	invision.ini
SECTION:	SSSCONFIG
PARAMETER	CREATE_CHAIN
VALUE(S):	boolean
DEFAULT	FALSE
SYNTAX:	CREATE_CHAIN = TRUE

Should the chain of all objects created at all? If you don't need this chain, it's just a waste of CPU power.

Command line argument

The Spreadsheet Server allows to provide a service name section as the server's command line parameter. This allows to specify a distinct publisher section upon program start-up using the command line parameter. This selection is echoed in the Spreadsheet Server status window after the lines of created records read from the database file sss_p.dat were shown.

Environment variable adjustments

In order to provide a simple switch in between different Spreadsheet Server installations the previous way of placing the dynamically loaded library "ssserver.dll" in the %WINDIR%\System32 directory is changed. Now, the library is placed in the Spreadsheet Server installation directory and has to be referenced from there.

The installation script adjusts the user part of the environment variable PATH to append the installation directory. If an earlier installation of the Spreadsheet Server is already defined in this PATH variable, it either has to be removed or placed behind the new installation path.

This PATH variable adjustment is made only for the user who installed the Spreadsheet Server. If other users need to run Excel accessing the Spreadsheet Server's dynamically linked library as well, their user PATH variable has to be extended in order to contain the installation path as well (**Start | Settings | Control Panel | System | Advanced | Environment Variables** and create a new PATH user variable - if it does not yet exist – and insert the Spreadsheet Server installation path or append a semicolon followed by the Spreadsheet Server installation path). Otherwise, a run-time error message telling

Run-time error '53':

File not found: SSSERVER.DLL

is shown.

If you see the above error message in spite of a proper entry in the user part of the PATH variable, check the output of the set command in a Windows cmd box (**Start | Run...**). If

the path extension does not show up for the PATH variable, please contact your system administrator.

Alternatively, a user having administrator rights on the computer system may include the installation path in the system's PATH variable, then it will be available for each user working on the system.

3 Publishing from a spreadsheet

See the samples in the *SamplesSimple.xls* and *SamplesDetailed.xls* files provided in the distribution.

At start-up the Spreadsheet Server will load file *sss_p.dat* which is held in the install directory. This file lists the names of the records that were published by the Spreadsheet Server on the last run. If this file does not exist the Spreadsheet Server will create it and add names of records as new ones are published.

As new records are encountered, the Spreadsheet Server will create them and publish the data so that users of InVision/Slingshot 2, i.e. +Arena or Java client, can request these records. As updates are sent from the spreadsheet to the Spreadsheet Server, the Spreadsheet Server will publish these updates so that the InVision/Slingshot 2 user receives real-time data.

Updates will only be published to existing records if there has been a change in any field values or a new field has been added to the set of fids.

If a new field is added or an existing field is updated with data of different length to the last data for it, the entire record is published.

All published objects are permanent, this means they will stay in the Publisher cache until the Spreadsheet Server is shut down.

There is no predefined set of fields that the user can publish for records. The user can add new fields to an existing record.

All the functions described below may be used directly in a cell of the spreadsheet or within a Visual Basic macro.

Publisher Functions

The Spreadsheet Server allows the user to publish records using the following functions:

1. Publish Rate (**PR**) Function
2. Publish Rate Extended (**PREx**) Function
3. Publish Rate Single (**PRE1**)) Function
4. Refresh Updates (**REFP**) Function

Publish Rate (PR) Function

A call to this function will instruct the server to send an update to *szRecord* specified in *szData*.

Public Declare Function PR Lib "ssserver.dll" (ByVal szRecord As String, ByVal szData As String) As Integer

where

szRecord	The name of the record that the user wishes to send data to
szData	This is a string of field IDs with their corresponding data fields. This string must have the following format... <field ID 1>,<field Data 1>,<field Id 2>,<field Data 2>,<field ID n>,<field Data n>

E.g. To update a bid and offer in record EUR= the user should call:

PR ("EUR=", "22,1.65,25,1.66")

Publish Rate Extended (PREx) Function

A call to this function will instruct the server to send an update to *szRecord* specified in *szData* which is delimited by *szDelimiter*.

Public Declare Function PREx Lib "ssserver.dll" (ByVal szRecord As String, ByVal szDelimiter As String, ByVal szData As String) As Integer

where

szRecord	The name of the record that the user wishes to send data to
szDelimiter	A single character delimiting the data
szData	This is a string of field IDs with their corresponding data fields. This string must have the following format: <field ID 1><D><field Data 1><D><field Id 2><D><field Data 2><D>.....<field ID n><D><field Data n> where <D> is the delimiter.

E.g. To update a bid and offer in record EUR= the user should call:

PREx ("EUR=", "#", "22#1.65#25#1.66")

Publish Rate Single (PRE1) Function

A call to this function will instruct the server to send an update of one field to *szRecord* specified in *szFieldID* and *szFieldData*.

Public Declare Function PRE1 Lib "ssserver.dll" (ByVal szRecord As String, ByVal szFieldID As String, ByVal szFieldData As String) As Integer

where

szRecord	The name of the record that the user wishes to send data to
szFieldID	The field ID (e.g. "22")
szData	The data for this field (e.g. "2.5612").

E.g. To update a bid and offer in record EUR= the user should call:

PRE1 ("EUR=", "22", "1.65")

Refresh Updates (REFP) Function

A call to this function will instruct the server to re-send an update to *szRecord*. The data that is sent in this case is the data held in the Spreadsheet Server cache.

Public Declare Function REFP Lib "ssserver.dll" (ByVal szRecord As String) As Integer

where

szRecord	The name of the record that the user wishes to send data to. If this is a
----------	---

	NULL string that the Spreadsheet Server will send updates for all cached records.
--	---

E.g. To refresh record EUR= the user should call:

REFP ("EUR=")

E.g. To refresh all records:

REFP ("")

Drop Record (DROPP) Function

A call to this function will instruct the server to drop the record *szRecord* and remove it from its internal cache. All InVision clients will be notified. It is possible to re-create the record immediately again by using the PR function – but better use the CLEARP function in this case.

Public Declare Function DROPP Lib "ssserver.dll" (ByVal szRecord As String) As Integer

where

szRecord	The name of the record that the user wishes to remove from cache.
----------	---

E.g. To remove record EUR= the user should call:

DROPP ("EUR=")

Clear Record (CLEARP) Function

A call to this function will instruct the server to remove all fields from the record *szRecord*, but the record will stay in the internal cache. All InVision clients will be notified. This is a good way to remove unwanted fields from a record. If a time field is configured, this field will stay and is updated with the actual time.

Public Declare Function CLEARP Lib "ssserver.dll" (ByVal szRecord As String) As Integer

where

szRecord	The name of the record that the user wishes to clear all fields within.
----------	---

E.g. To clear all fields in record EUR= the user should call:

CLEARP ("EUR=")

Error Handling

Calls to any of the functions **PR**, **PRx**, **PRx**, **REFP**, **DROPP**, **CLEARP** will return values as follows:

Error Code	Description
0	Operation successful
-1	Operation failed

-2	Spreadsheet server is not running
----	-----------------------------------

The following are reasons for the failure of publishing an update:

- Spreadsheet Server is not set up for publishing.
- No record name is specified.
- Failure to allocate memory.
- The call to **InvCreateObject()** failed, probably because have published the maximum number of records.
- The call to **InvPublishObject()** failed.

The Spreadsheet Server runs with the standard monitor i.e. generates log files. The user should look here for more error detail.

Using the functions

To make the function **PR, PRex, PRx, REFP, DROPP, CLEARP** available to all worksheets with in your Excel workbook perform the following steps:

Open a new workbook file.

Using the Tools | Macro | Visual Basic Editor command, which displays the Visual Basic Editor, select the Insert | Module command and insert the following function declarations into the new module:

Public Declare Function PR Lib "ssserver.dll" (ByVal szRecord As String, ByVal szData As String) As Integer

Public Declare Function PRex Lib "ssserver.dll" (ByVal szRecord As String, ByVal szDelimiter As String, ByVal szData As String) As Integer

Public Declare Function PR1 Lib "ssserver.dll" (ByVal szRecord As String, ByVal szFieldID As String, ByVal szFieldData As String) As Integer

Public Declare Function REFP Lib "ssserver.dll" (ByVal szRecord As String) As Integer

Public Declare Function DROPP Lib "ssserver.dll" (ByVal szRecord As String) As Integer

Public Declare Function CLEARP Lib "ssserver.dll" (ByVal szRecord As String) As Integer

4 Contributing from a Spreadsheet

Contribution Functions

The **CR**, **CREx**, **CRE1** and **REFC** functions allow the user to contribute data to records as an InVision client.

Contribute Rate (CR) Function

A call to this function will instruct the server to send an insert, i.e. contribution to *RecordName* as specified in *szData*. Note that the server will send this insert to all servers that it has logged onto.

Public Declare Function CR Lib "SSSERVER.DLL" (ByVal szRecord As String, ByVal szData As String) As Integer

where:

szRecordName	This is the name of the record that the user wishes to contribute data to.
szData	This is a string of field IDs with their corresponding data fields. This string must have the following format: <field ID 1>,<field Data 1>,<field ID 2>,<field Data 2>,<field ID n>,<field Data n>

Contribute Rate Extended (CREx) Function

A call to this function will instruct the server to send an insert, i.e. contribution to *szRecord* as specified in *szData* which is delimited by *szDelimiter*. Note that the server will send this insert to all servers that it has logged onto.

Public Declare Function CREx Lib "SSSERVER.DLL" (ByVal szRecord As String, ByVal szDelimiter As String, ByVal szData As String) As Integer

where:

szRecordName	This is the name of the record that the user wishes to contribute data to.
szDelimiter	A single character delimiting the data
szData	This is a string of field IDs with their corresponding data fields. This string must have the following format: <field ID 1>,<field Data 1>,<field ID 2>,<field Data 2>,<field ID n>,<field Data n> where <D> is the delimiter.

Contribute Rate Single (CRE1) Function

A call to this function will instruct the server to send an insert, i.e. contribution to a single field in *szRecord* as specified in *szFieldID* and *szFieldData*. Note that the server will send this insert to all servers that it has logged onto.

Public Declare Function CRE1 Lib "SSSERVER.DLL" (ByVal szRecord As String, ByVal szFieldID As String, ByVal szFieldData As String) As Integer

where:

szRecordName	This is the name of the record that the user wishes to contribute data to.
szFieldID	The field ID (e.g. "22")
szFieldData	The field data (e.g. "2.6354")

Refresh Contributions (REFC) Function

A call to this function will instruct the server to re-send an insert, i.e. contribution to *szRecord*. The data that is sent in this case is the data held in the server cache. If the object is not held in cache nothing is sent.

Public Declare Function REFC Lib "SSSERVER.DLL" (ByVal szRecord As String) As Integer

where:

szRecordName	This is the name of the record that the user wishes to resend the cached contribution for. If this is a NULL string then all cached contributions are sent.
--------------	---

Error Handling

Calls to any of the functions **CR**, **CREx**, **CRE1**, **REFC** give return values as follows:

Error Code	Description
0	Operation successful
-1	Operation failed
-2	Spreadsheet server is not running

The following are reasons for the failure of a contribution:

- No record name is specified.
- Failure to allocate memory.
- No connection to a contribution server is available.

Using the functions

To make the function **CR**, **CREx**, **CRE1** and **REFC** available to all worksheets with in your Excel workbook perform the following steps:

1. Open a new workbook file.

2. Using the Tools | Macro | Visual Basic Editor command, which displays the Visual Basic Editor, select the Insert | Module command and insert the following function declarations into the new module:

Public Declare Function CR Lib "SSSERVER.DLL" (ByVal szRecord As String, ByVal szData As String) As Integer

Public Declare Function CREx Lib "SSSERVER.DLL" (ByVal szRecord As String, ByVal szDelimiter As String, ByVal szData As String) As Integer

Public Declare Function CRE1 Lib "SSSERVER.DLL" (ByVal szRecord As String, ByVal szFieldID As String, ByVal szFieldData As String) As Integer

Public Declare Function REFC Lib "SSSERVER.DLL" (ByVal szRecord As String) As Integer

5 Example 1

Setup a spreadsheet as shown below.

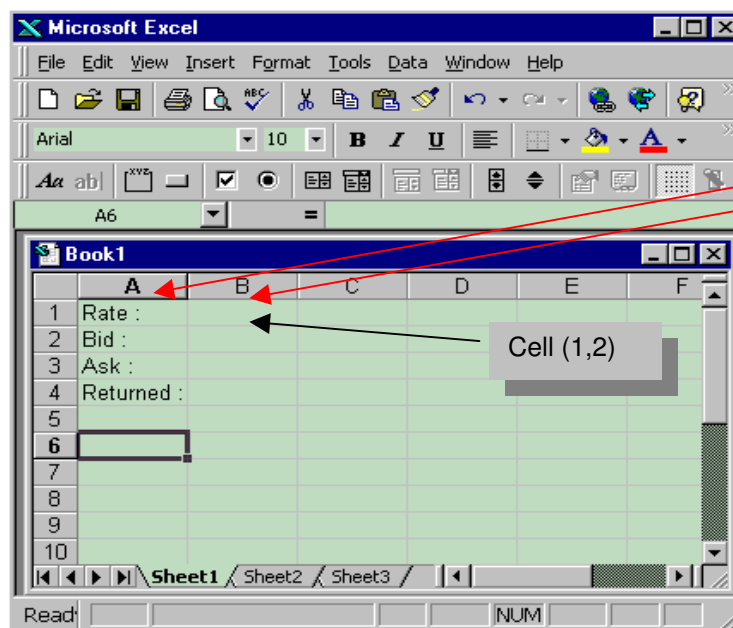
Note:

The cell (1,2) should be free to store the rate name.

The cell (2,2) should be free to store the bid value.

The cell (3,2) should be free to store the ask value.

The cell (4,2) should be free to store the Spreadsheet server return value.



A = 1, B = 2,
C=3 etc.

Cell (1,2)

Place a button on the sheet and connect the following code to the button.

```
Sub Button1_Click()  
' Below is the declaration of variables to be used in this macro function  
Dim szBid As String  
Dim szAsk As String  
Dim szRecord As String  
Dim szFids As String  
Dim iReturn As Integer
```

Name of the Macro assigned to the newly added button. Notice that Excel automatically generate the macro function based on the button's name (named Button1, Button2 .. ButtonN) and the action performed on the button (Click).

```
' Copy the rate name, bid value and ask value from the spreadsheet into the various variable  
szRecord = Cells(1, 2).Value  
szBid = Cells(2, 2).Value
```

szAsk = Cells(3, 2).Value

' Create a fid stream, containing the bid and the ask fids

' this will result in the fid stream

' 22,(contents of bid cell),25,(contents of ask fid)

szFids = "22," & szBid & ",25," & szAsk

' Call the publish rate (PR) function, passing the record name and the fid stream to it.

iReturn = PR(szRate, szFids)

' Place the RETURN VALUE in the returned cell on the spreadsheet.

Cells(4, 2).Value = iReturn

End Sub

6 Example 2

To publish market data from a Microsoft Excel spreadsheet, follow these steps:

Open a new workbook file.

Using the Tools | Macro | Visual Basic Editor command displays the Visual basic Editor, select the Insert | Module command and insert the following function declaration into the module:

Public Declare Function PR Lib "ssserver.dll" (ByVal szRecord As String, ByVal szData As String) As Integer

This declaration makes the **PR** function available to all worksheets within your workbook.

Decide how many fields (columns) you want to publish, and enter a unique numerical code for each one. For example, 1 for Bid, 2 for Ask and 3 for Time.

You can use the conventional FID naming scheme when assigning the field code to each field – or you can enter a number which you choose yourself. All that matters is that the number is unique and that it corresponds with the number specified for that field in the DataSource parameter of the SLS file.

Decide how many records (rows) you want to create, and enter a label name for each one. For example, GBP, DEM, IEP, FRF, JPY and ITL.

See the sample array of rows and columns below.

	A	B	C	D
1		1	2	3
2	GBP			
3	DEM			
4	IEP			
5	FRF			
6	JPY			
7	ITL			

Populate the cells under the fields labels with data. You can do this in three ways:

- Enter an actual value in a cell.
- Enter a reference to another spreadsheet cell which contains an actual value – for example, the result of a calculation.
- Enter a DDE link which references a data value in another Microsoft Windows application.

To the right of the last column of data (in this case, the Ask column), add a new label – called Function, for example – to indicate that the rightmost row is to be used for holding the public record function for each row in the array. While functions can be located in any cell on the spreadsheet, it is tidier to locate them in a single column.

For each record in the array, enter the function in the following format:

=PR(cell address of record label; field ID of first field; cell address of first field; field ID of second field; cell address of second field;...)

In summary, the parameters within the Publish Record function take the form of the cell address of the record label, followed by the IDs and cell addresses of the data fields in the row.

In this example, the Function column would read as follows:

Function
=PR(A2; B1 & ";" & B2 & ";" & C1 & ";" & C2 & ";" & D1 & ";" & D2)
=PR(A2; B1 & ";" & B3 & ";" & C1 & ";" & C3 & ";" & D1 & ";" & D3)
=PR(A2; B1 & ";" & B4 & ";" & C1 & ";" & C4 & ";" & D1 & ";" & D4)
=PR(A2; B1 & ";" & B5 & ";" & C1 & ";" & C5 & ";" & D1 & ";" & D5)
=PR(A2; B1 & ";" & B6 & ";" & C1 & ";" & C6 & ";" & D1 & ";" & D6)
=PR(A2; B1 & ";" & B7 & ";" & C1 & ";" & C7 & ";" & D1 & ";" & D7)

All fields must be specified each time that the **PR** function is called.

7 Example 3 - The Chains Sheet

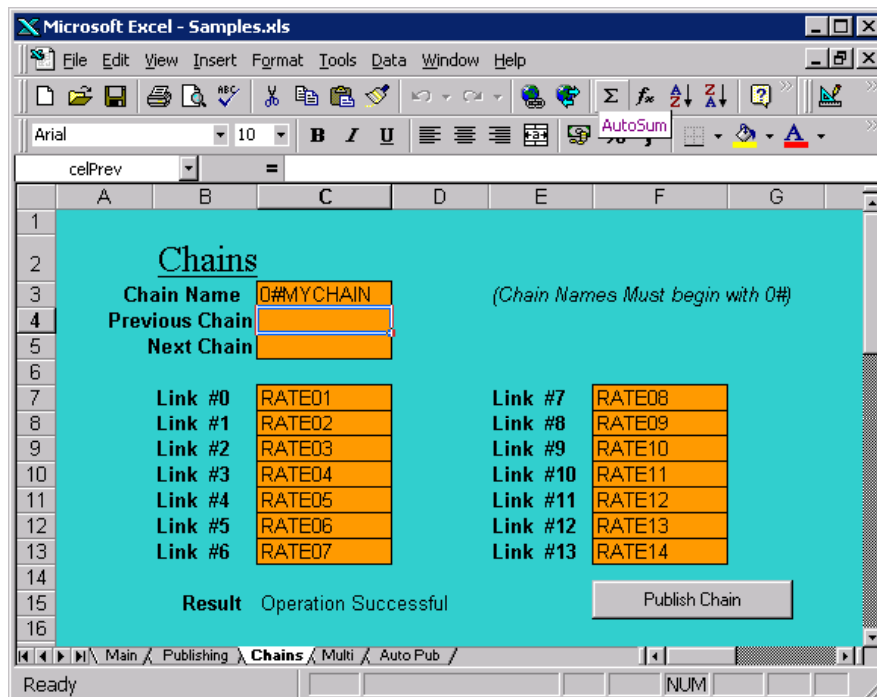
Publishing a chain is similar to publishing a record, the same function (**PR**) is used. The Spreadsheet Server distinguishes between a record and a chain through the application of a standard format for a chain name. All initial chain segments provided to the Spreadsheet Server should begin with "0#", following chain segments usually begin with increased decimal numbers ("1#", "2#", "3#", "4#", "5#", "6#", "7#", "8#", "9#", "10#", "11#" ...).

In publishing a chain a predefined set of fields are published, these are fields 237 to 253 inclusive. Each field has a specific purpose, recognised by the workstation application used to request the chain. When a workstation requests a chain, the Spreadsheet Server returns the chain name, the previous chain link name and the next chain link name. It also returns the number of elements in this link and then (up to 14) elements. Each element consists of a record name. The workstation application then retrieves each rates data individually from the Spreadsheet Server and displays them in the chain format to the user.

The field ID specifications for a chain are as follows:

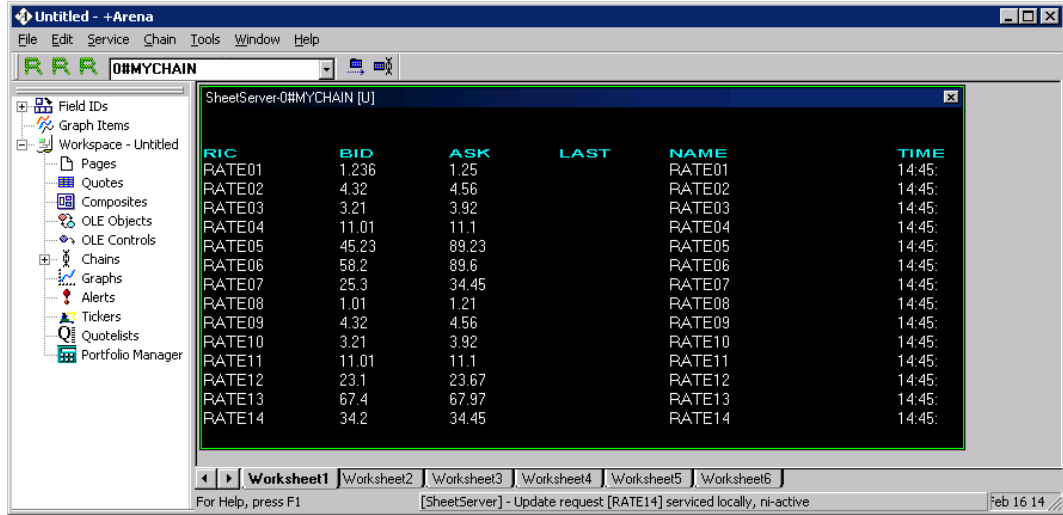
FID Number	Description
237	Previous Chain Link Name
238	Next Chain Link Name
239	Number of Elements in this Link
240	Element number 0 (e.g. RATE1=)
:	
253	Element number 13 (e.g. (RATE14=)

The chains sheet is shown as follows:



All the code behind this sheet can be found in the sample Excel sheet **SamplesDetailed.xls** provided.

The picture below shows the chain in +Arena:



The picture below shows chain elements displayed as a quotelist under Slingshot 2:

Slingshot from Swissrisk. Providing solutions for financial markets. - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites

Address <http://127.0.0.1:8080/examples/SSServer.html> Go Links

Home Contact WDS Monitor Demo Index

swissrisk
Financial Systems

the speed you need!

SLINGSHOT

Spread Sheet Server

The Spreadsheet Server allows records to be published over Slingshots Web distributions system. Records are entered through the use of excel spreadsheets and the data is placed on the relevant service through the use of Windows shared memory and the Spreadsheet Server applications. As the Spreadsheet Server application communicates with the Slingshot Web Distribution Systems, the Spreadsheet Server application may act as an independent publisher, to the Java client.

SpreadSheet Server Example

	BID	ASK	DEAL	SRCE
SFS.EUR=	0.9875	0.9881	SSServer	Swissrisk
SFS.JPY=	119.84	119.89	SSServer	Swissrisk
SFS.GBP=	1.5308	1.5314	SSServer	Swissrisk
SFS.CHF=	1.4848	1.4853	SSServer	Swissrisk
SFS.XAU=	311.25	311.76	SSServer	Swissrisk
SFS.XAG=	4.8	4.82	SSServer	Swissrisk
SFS.AUD=	0.5602	0.5607	SSServer	Swissrisk
SFS.CAD=	1.5172	1.5182	SSServer	Swissrisk
SFS.SEK=	9.1691	9.177	SSServer	Swissrisk

	BID	ASK	DEAL	SRCE
SFS.NOK=	7.4975	7.5007	SSServer	Swissrisk
SFS.DKK=	7.5008	7.5041	SSServer	Swissrisk
SFS.RUB=	31.512	31.522	SSServer	Swissrisk
SFS.TRL=	1577000	4564	SSServer	Swissrisk
SFS.ISK=	86.74	86.88	SSServer	Swissrisk
SFS.PLN=	4.049	4.053	SSServer	Swissrisk
SFS.CZK=	29.82	29.986	SSServer	Swissrisk
SFS.HUF=	246.44	246.84	SSServer	Swissrisk
SFS.UAH=	5.328	5.3288	SSServer	Swissrisk

Applet net.Slingshot.Viewer.Viewer started

Internet

A Support Information

Should you experience any problems when using the program, please contact your system administrator first.

You can also contact the Swissrisk support department. It can be reached via hotline@swissrisk.com

Phone: +49 (0)69 50952-111

Fax: +49 (0)69 50952-199

For more information on Swissrisk products and services see our web pages:

<http://www.swissrisk.com>

Index

+Arena 6, 14, 26
environment variable
 PATH 12

Function

CLEARP 16, 17
CR 18, 19
CRE1 19
CREx 18, 19
DROPP 16, 17
PR 7, 11, 14, 16, 17, 22, 23, 25
PRE1 14, 15, 16, 17
PREx 14, 15, 16, 17
REFC 19
REFP 14, 15, 16, 17, 19
RP 18

IAW 6

invision.ini 8, 11

Parameter

ASK_FID 10
BID_FID 10

CHAIN_BASENAME 12
CONTRIB_USERNAME 9
CONTRIB_USERNAME_FID 10
CONTRIBUTION 9
CREATE_CHAIN 12
DELIMITER 9
MASK 9
PERM_FID 11
PRELOAD_STALE 11
PUBLISHER_CACHE_FILE 11
SEND_TIME 8
TIME_FID 8
TIME_FORMAT 8
PATH 12
PE number 10
Restrictions 7
run-time error 13
Slingshot 2 6, 14, 26
sss_p.dat 11, 12, 14
SSServer.dll 12