OpenDeveloper Sorting methods

This guide will outline the five new methods created for sorting events and saving a user defined sortID. The format necessary for the methods is a [1:n] vector matrix which, when passed to the SaveSortCodes method, will associate each index and its paired neighbor as a set of an event index and sort code.

For example, the vector [5, 2, 6, 1, 7, 2, 8, 1] will have four event indexes 5,6,7,8 and four sort codes 2,1,2,1 when passed the method will classify these indexes with their paired sort codes and save them to a user defined sortID.

VARIANT GetEvTsqIdx();

This method is used to return an index array in long which contains how the events are distributed in the .tsq file. This method should be called immediately after the ReadEventsV. The options parameter in the ReadEventsV call MUST be set to “IDXPSQ” in order for the GetEvTsqIdx method to return the correct index array.

Example:

```
TT.ConnectServer('Local', 'Me')

TT.OpenTank('SomeTank', 'R')

TT.SelectBlock('Block-1')

NumRecs=TT.ReadEventsV(1000,'Snip',1,0,0,0,’IDXPSQ’)

IndexArray=TT.GetEvTsqIdx % the index array returned should contain the same
                              number of values as the ReadEventsV call.
```

BOOL SaveSortCodes(BSTR SortName, BSTR snipName, long idxChan, BSTR sortCondition, VARIANT sortCodeArray);

This method saves all sorting information to a user defined sortID. The SortName parameter accepts a string that will define the name of the newly created sortID. The snipName parameter is the name of the event in 4 character string form. The idxChan parameter defines the target channel that the sortID saves the events + sort codes to. The sortCondition parameter is a string that allows the user to define any sortConditions (such as the algorithm used) and can later be retrieved using the method GetSortCondition after this value has been initialized. Finally, the sortCodeArray parameter is a vector that contains each event index and its paired sortcode as described above. An example of this method is provided in the .m file

BSTR GetSortCondition(BSTR sortName, BSTR snipName, long idxChan);

This method returns the defined sort condition associated with the specified sortID. The sortName parameter accepts a string that will specify the desired sortID for which a sort condition will be returned. The snipName parameter is the name of the event in 4 character string form. The idxChan parameter defines the target channel that the desired sortID will return a sort condition for. When called before any sort conditions are set, this method will return a ‘’ (null) string. After the SaveSortCodes method is called, the GetSortCondition method will return the string set from the sortCondition parameter in the SaveSortCodes method (see method for details).

BOOL DeleteSortCode(BSTR sortName, BSTR snipName, long idxChan);

This method is used to delete a single channel’s sort codes from the desired sortID. This is equivalent to right-clicking a sorted channel in OpenSorter, hitting Delete and removing the check from that channel. Note that only individual channels may be deleted per call of this method and no sortID can be deleted either. The sortName parameter specifies the which sortID to use while the snipName and idxChan parameters specify which tank event and subsequent channel. You may use the GetSortChanMap method to verify that the target channel’s sort codes were indeed removed.

VARIANT GetSortChanMap(BSTR sortName, BSTR snipName);

This method returns a [1:1024] vector matrix which indicates which channel(s) of the specified sortID and event name are sorted (1) or unsorted (0). Note that this vector matrix begins its index at channel 0 which does not exist. You may format the returned vector matrix to exclude the first entry or simply just ignore it. If this method is called after a DeleteSortCode call to a specified channel, calling this method afterward on the same sortID will return a 0 for that channel’s index (again remember that the first index is disregarded).

Example:

Values returned

```
[ 0    1    1    0    0    1 . . .
```

This would indicate that channels 1, 2, and 5 have been sorted and channels 3 and 4 have not been sorted