SIMPLEXLS – User guide

Thank you for buying SIMPLEXLS. Please, study the user guide carefully. In case you still have any questions, or you find any bugs don't hesitate to write an e-mail (error@lonesoft.hu) and we will solve the problem as soon as it is possible.

Basic functions

SIMPLEXLS class can convert grouped and formatted XLS file by using also SUM function based on any kind of table, cursor or view. It makes the converting quicker and the result excel file is more user friendly than other applications.

Lets see, how easy it can be:

```
SET CLASSLIB TO simplexls
*** progress bar class
SET CLASSLIB TO alap additive

SELECT 0
USE (HOME()+"samples\northwind\orders.dbf")
oSimpleXLS = CREATEOBJECT("simplexls")
oSimpleXLS.SimpleXLS
```

Then the process begins, which you can follow on a progress bar. By pressing ESC you can quit.

Parameters

BOLD

Comma separated list of bold formatted field(s):

```
oSimpleXLS.bold = "customerid,shipname"
```

DETAIL

Comma separated list of detail field(s), if empty all fields appearing :

```
oSimpleXLS.detail = "orderdate,requireddate,shippeddate,freight"
```

FILTERDESC

Information text about the filter conditions, appears on the top of XLS:

```
oSimpleXLS.filterDesc = "All records"
```

FIRM

User name:

```
oSimpleXLS.filterDesc = " Northwind Inc."
```

GETCAPTION

The software tries to fill the headers with the database caption property to make it more understandable. If you know that the source does not contains such property – e.g. in case it is a cursor - you should keep it "false" to ensure quicker converting, otherwise set it "true".

```
oSimpleXLS.getcaption = .T.
```

GROUPBACKCOLOR

Background color of the group header and group footer sections.

```
oSimpleXLS.groupbackcolor = "255,255,0"
```

GROUPBY

Comma separated list of group by field(s):

```
oSimpleXLS.groupby = "customerid"
```

GROUPHEADER

Comma separated list of those field(s), which should be appear in the group header, but not groupby conditions (therefore not listed in the groupby property):

```
oSimpleXLS.groupheader = "shipname,shipcountry"
```

HEADERBACKCOLOR

Background color of the header and footer of the list.

```
oSimpleXLS.headerbackcolor = "255,0,0"
```

ITALIC

Comma separated list of italic formatted field(s):

```
oSimpleXLS.italic = "requireddate"
```

MYCAPTION

Title of the Excel sheet:

```
oSimpleXLS.mycaption = "Sample list"
```

MYDATASOURCE

The source table or cursor name, if empty working on the selected alias:

```
oSimpleXLS.mydatasource = "customers"
```

RIGHTORDERED

Set it "true", if your source is already ordered according to the group by fields. This makes the converting quicker, as the program reorders the source data by the groupby property for the proper result. (because default rightordered = .f.)

```
oSimpleXLS.rightordered = .t.
```

SUM

Comma separated list of summed field(s). The program uses SUM function of Excel, so if the user changes a number on the sheet, the whole workbook will be recalculated. This kind of Excel table is more easy to use than those converted from frx.

```
oSimpleXLS.sum = "freight"
```

UNDERLINE

Comma separated list of underlined field(s):

```
oSimpleXLS.italic = "requireddate"
```

XLSNAME

If empty, the result Excel will appear. If filled with a proper filename, Excel will be saved but will not appear.

```
oSimpleXLS.XLSName = "c:\sample.xls"
```

Let see the code that created sample.xls:

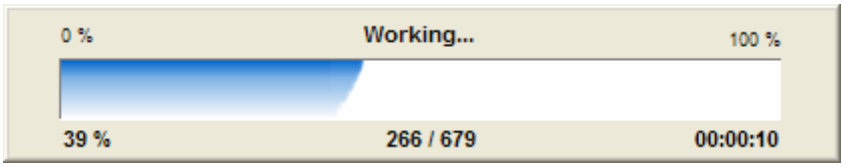
```
SET CLASSLIB TO simplexls
SET CLASSLIB TO alap additive

SELECT 0
USE (HOME()+"samples\northwind\orders.dbf")
oSimpleXLS = CREATEOBJECT("simplexls")
oSimpleXLS.getcaption = .t.
oSimpleXLS.detail = "orderdate,requireddate,shippeddate,freight"
oSimpleXLS.groupby = "customerid"
oSimpleXLS.groupheader = "shipname,shipcountry"
oSimpleXLS.bold = "shipname"
oSimpleXLS.italic = "requireddate"
oSimpleXLS.sum = "freight"
oSimpleXLS.SimpleXLS
```

In case of huge databases ignoring formatting and/or setting the background color white "255,255,255" makes the process quicker.

Using processbar class

You can use processbar class for your own applications to indicate a status of a procedure.




```
SET CLASSLIB TO alap additive
SELE (THIS.mydatasource)
COUNT FOR NOT DELETED() TO m.nMax_
*** Initializing process bar
oProcessbar = CREATEOBJECT("processbar")
oProcessbar.label3.CAPTION = "Working..."
oProcessbar.VISIBLE = .T.
**** other events
oProcessbar.nMax = m.nMax_+ 3
oProcessbar.Fast = .T.
oProcessbar.STATUS = .T.
oProcessbar.nCurrent = 0
.
.
.
oProcessbar.nCurrent = oProcessbar.nCurrent+1
```

Set the number of steps of the procedure in the oProcessbar.nMax (initial steps + record number + closing steps). After every step increase the value of the oProcessbar.nCurrent. The nCurrent_assign method moves the processbar, if it gets to nMax closes the object.

Parameters

FAST

Moves the graphical indicator cracked (fast) or smooth (slow).

STATUS

Show percent, max item, current item, elapsed time or only the graphical indicator.