MDSheet Manual

About

MDSheet is a framework for model-driven spreadsheet development. This framework is provided as an extension for LibreOffice, which enables users to specify models for their spreadsheets from the spreadsheets themselves.

The MDSheet extension is an academic prototype, thus not ready for production environments. Moreover, its focus is on the implementation of model-driven techniques for spreadsheet development, leaving out features that are essential for end-users, e.g., “undo”.

Download

The MDSheet extension for LibreOffice is available here:

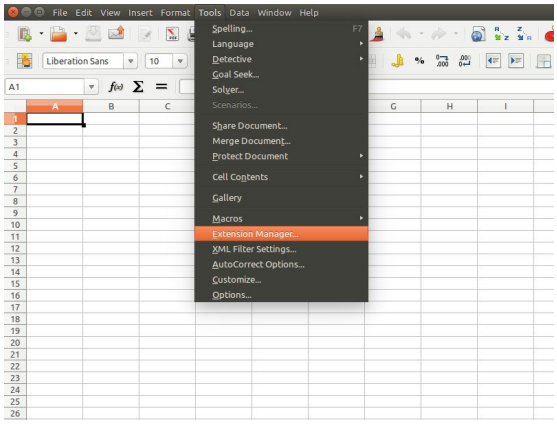
- http://alfa.di.uminho.pt/~jorgemendes/software/mdsheet/latest/MDSheet-Basic.oxt
- http://alfa.di.uminho.pt/~jorgemendes/software/mdsheet/latest/MDSheet.oxt

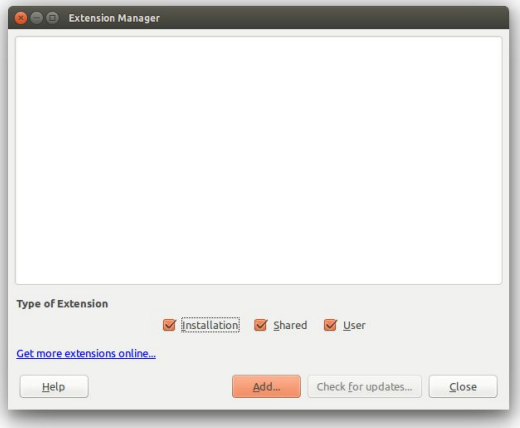
It is currently composed by two components:

- MDSheet-Basic – the StarBasic code for the interface; and,
- MDSheet – the backend system.

Installation

After downloading it, the extension can be installed from within LibreOffice, via the Extension Manager (Tools > Extension Manager).





From within the Extension Manager, press “Add…” and select the “MDSheet-Basic.oxt” file. Repeat the process for “MDSheet.oxt”.

The MDSheet extension can also be downloaded and installed from the command line:

```
# set the base URL in an environment variable for readability purposes
LATEST=http://alfa.di.uminho.pt/~jorgemendes/software/mdsheet/latest
# download the extension packages
wget $LATEST/MDSheet-Basic.oxt
wget $LATEST/MDSheet.oxt
# install the extension packages
unopkg add MDSheet-Basic.oxt
unopkg add MDSheet.oxt
```

Compilation

These steps were tested with Ubuntu 14.04 32bits. It should work with other GNU/Linux 32bits systems without major changes. Other operating systems and CPU architectures are not supported.

Get Needed Tools to Compile the Code

```
# install software from the software center
sudo apt-get install zip git \
  build-essential libmakefile-parser-perl \
  haskell-platform libreoffice-dev
```

```
# update cabal to a more recent version
cabal update
cabal install cabal-install

# include cabal's binaries path
echo "PATH=$HOME/.cabal/bin:$PATH" >> .bashrc
source .bashrc

# install other Haskell-related dependencies
cabal install cpphs
```

Get the Source Code

```
git clone git@bitbucket.org:SSaaPP/mdsheet-utils.git
git clone git@bitbucket.org:SSaaPP/2ota.git
git clone git@bitbucket.org:SSaaPP/mdsheet.git
git clone git@bitbucket.org:SSaaPP/mdsheet-libreoffice.git
```

Set Everything for Compilation

```
cd mdsheet-libreoffice/
# create a sandbox so there is no need to install
# this at the user/system level
cabal sandbox init
# add local dependencies
cabal sandbox add-source ../mdsheet-utils/
cabal sandbox add-source ../2ota/
cabal sandbox add-source ../mdsheet/
# install all dependencies (both local and from Hackage)
cabal install --only-dependencies
```

Configure

Some system specific settings have to be set by hand. For this, a Perl script tries to figure out which are the current settings, but users have the freedom to set their own settings. These settings are mainly some required paths with header files for compilation. Note that some Perl-related warnings may arise but can be safely ignored.

```
./configure
Path to Office installation? [/usr/lib/libreoffice]
Path to Office SDK installation? [/usr/lib/libreoffice/sdk]
Path to Office headers? [/usr/include/libreoffice]
Path to URE installation? [/usr/lib/ure]
Path to GHC headers? [/usr/lib/ghc/include]
Path to C++ headers? [/usr/include/c++/4.8]
```

Build

```
make
```

Utility Commands

Install (only MDSheet.oxt):

```
make install
```

Uninstall (only MDSheet.oxt):

```
make uninstall
```

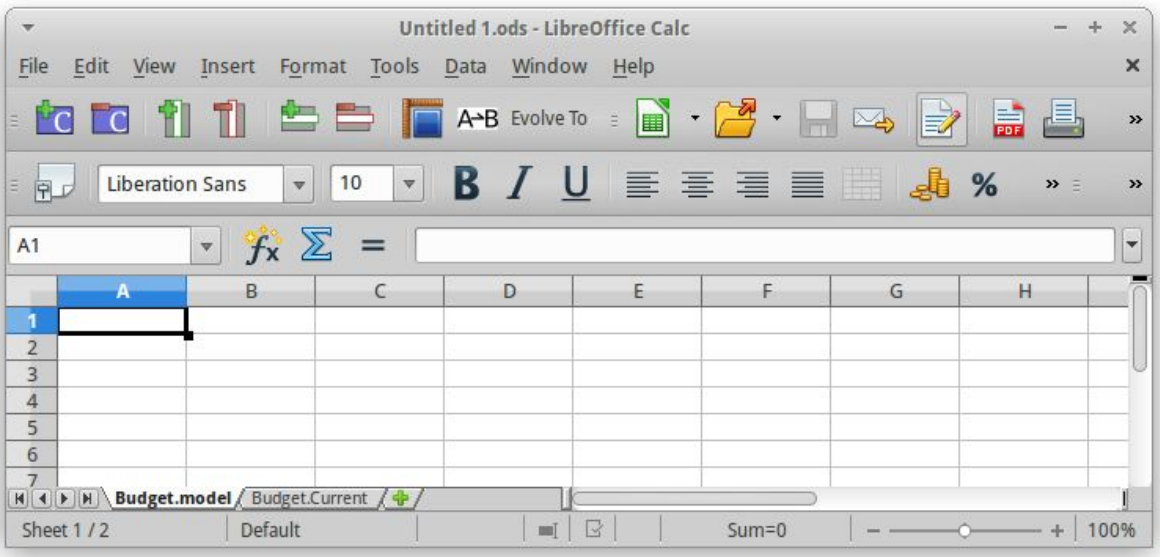
Run LibreOffice Calc without recovering files when errors occur:

```
make run
```

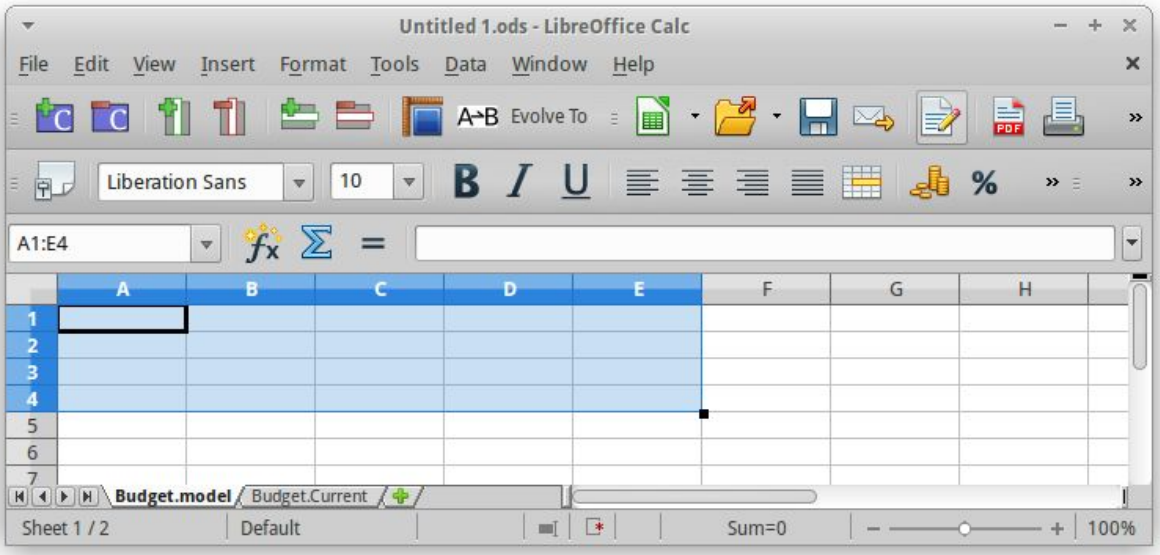
Usage

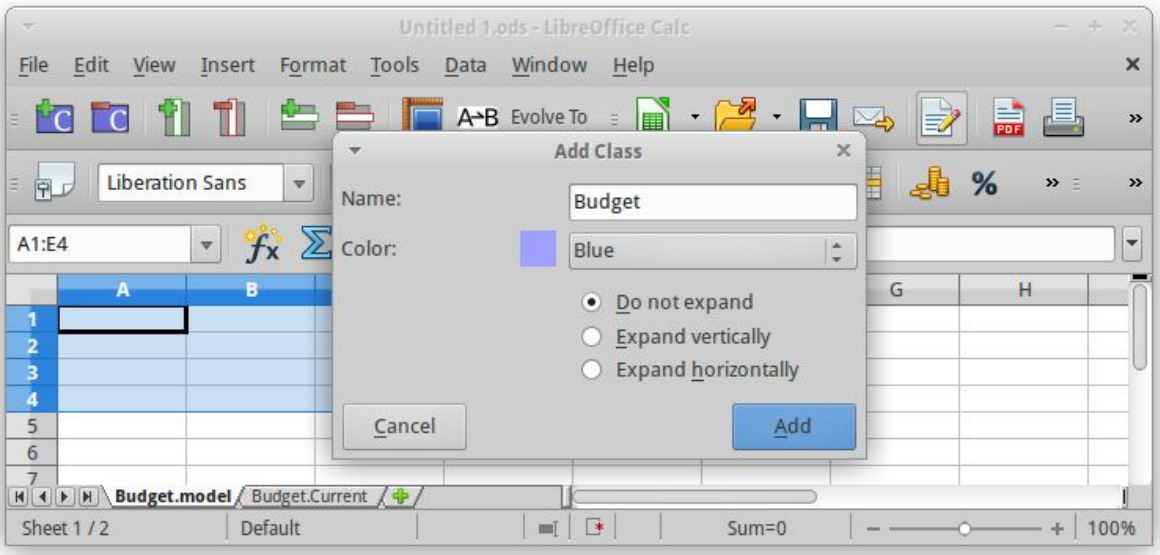
Model Creation

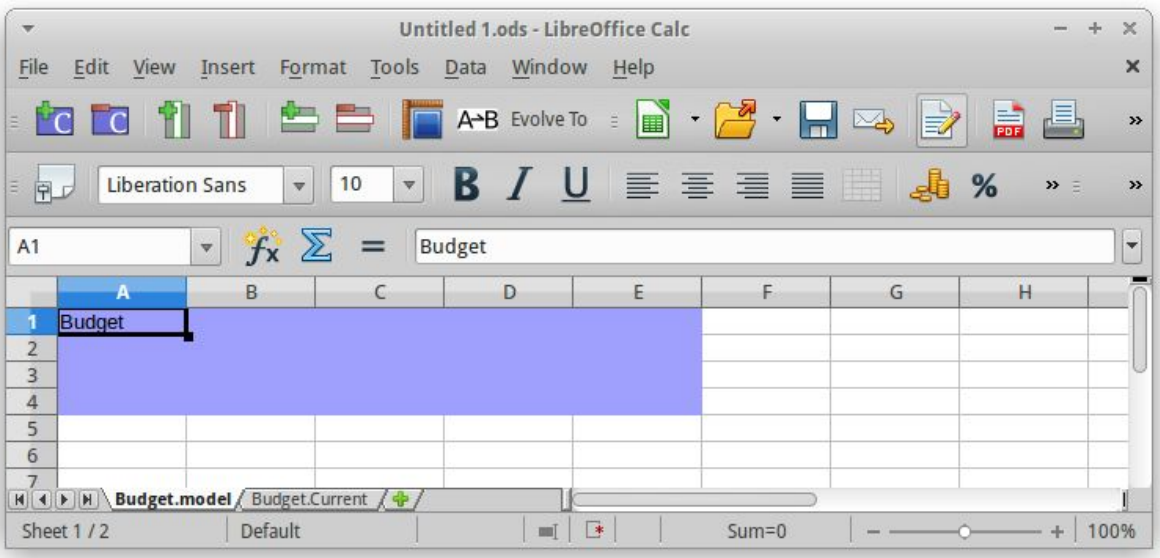
Create a new spreadsheet file and within that file create two worksheets. The name of one of the worksheets must end in ".model" (e.g., for a model named "Budget", the worksheet name is "Budget.model"), and the other one must start with the name name of the model and the name of the instance with a dot connecting them (e.g., for an instance of the model "Budget" named "Current", the worksheet name is "Budget.Current").



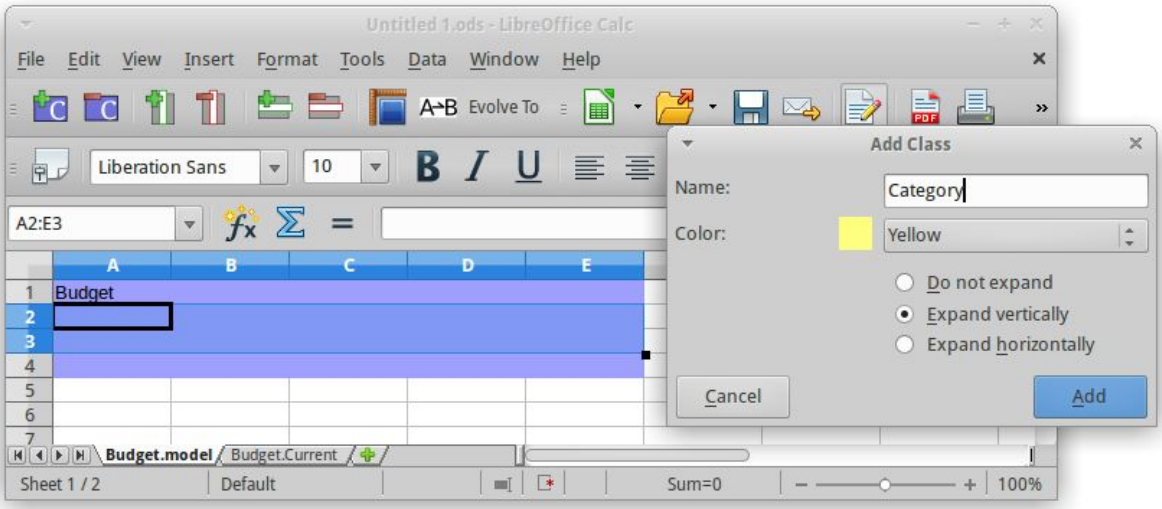
Then, a base class is needed to contain all the other classes and attributes. To create it, select the area that the class should use and press the "Add Class" ( ) button. Set the name of the class, select its color, and its expansion type. In the example below, a new class named *Budget* is added, it will be highlighted in *blue*, and the class will not expand.

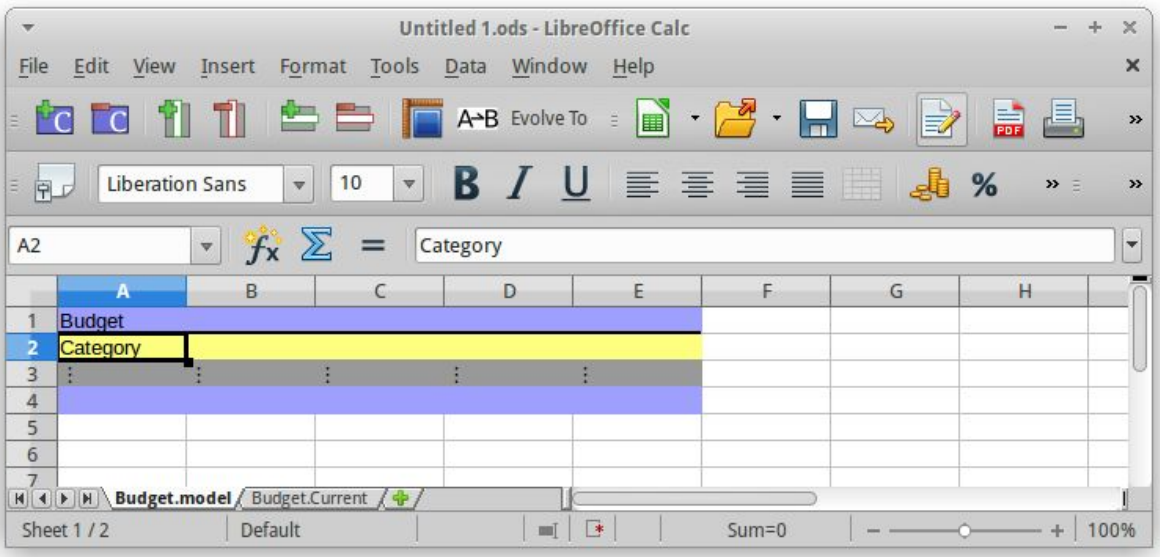

Untitled 1.ods - LibreOffice Calc
File Edit View Insert Format Tools Data Window Help
Evolve To
Liberation Sans
10
A1:E4
A B C D E F G H
1 2 3 4 5 6 7
Budget.model Budget.Current
Sheet 1 / 2
Default
Sum=0
100%


Untitled 1.ods - LibreOffice Calc
File Edit View Insert Format Tools Data Window Help
Evolve To
Liberation Sans
A1:E4
Add Class
Name:
Budget
Color:
Blue
Do not expand
Expand vertically
Expand horizontally
Cancel
Add
Budget.model Budget.Current
Sheet 1 / 2
Default
Sum=0
100%

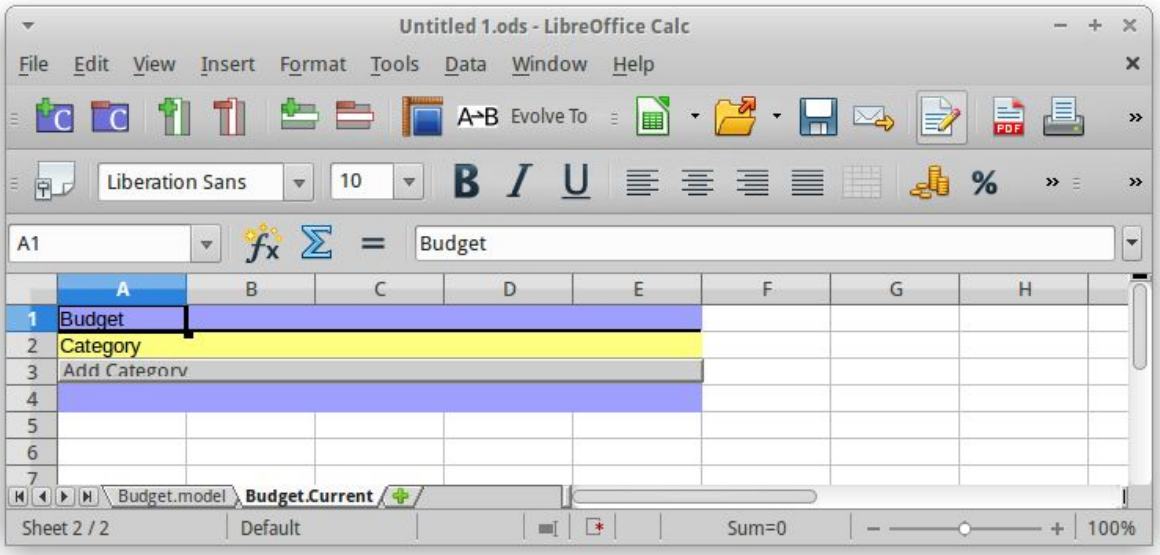

Untitled 1.ods - LibreOffice Calc
File Edit View Insert Format Tools Data Window Help
Evolve To
Liberation Sans
10
A1
Budget
A B C D E F G H
1 2 3 4 5 6 7
Budget
Budget.model Budget.Current
Sheet 1 / 2
Default
Sum=0
100%

Then, other classes can be added. The classes must be either of the same width or of the same height of the base class. This rules ensures that no unwanted deformations will occur. Furthermore, if a class is of the same width (height) of the base class, a row (column) belonging to the outer class must be present above (on the left) and below (on the right) of the new class (see the tiling rules in the ClassSheet specification).

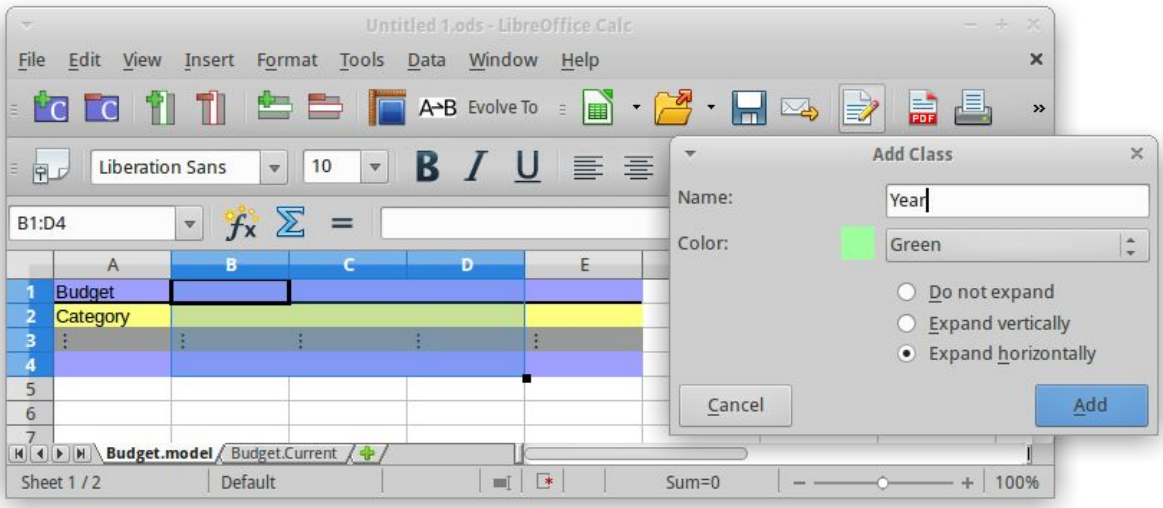




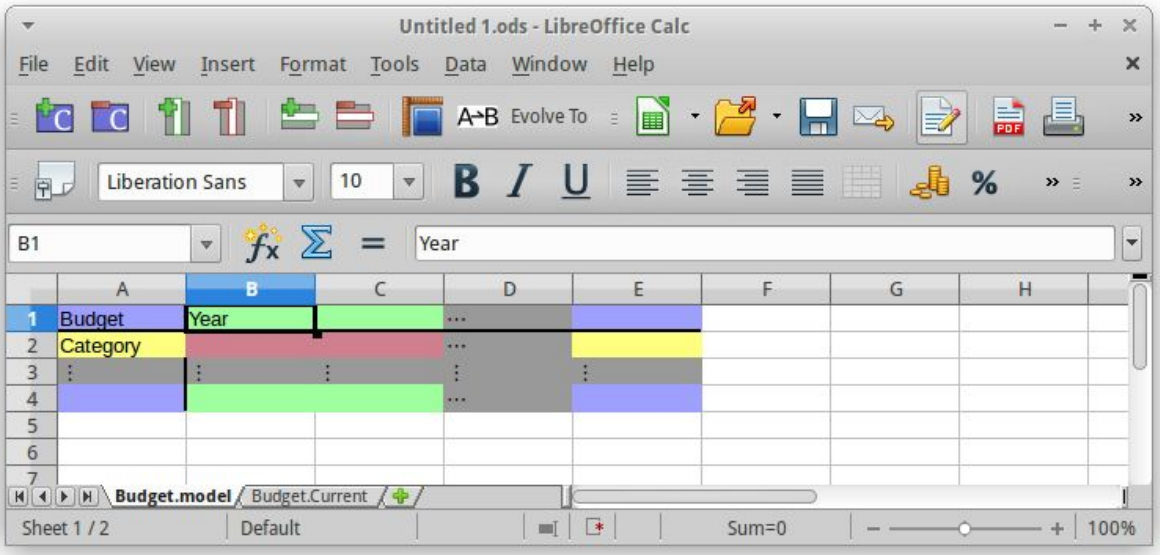
When an expandable class is added to the model, the same class is added to the instance, with the addition of a button that enables users to add instances of that class, as shown in the image below.



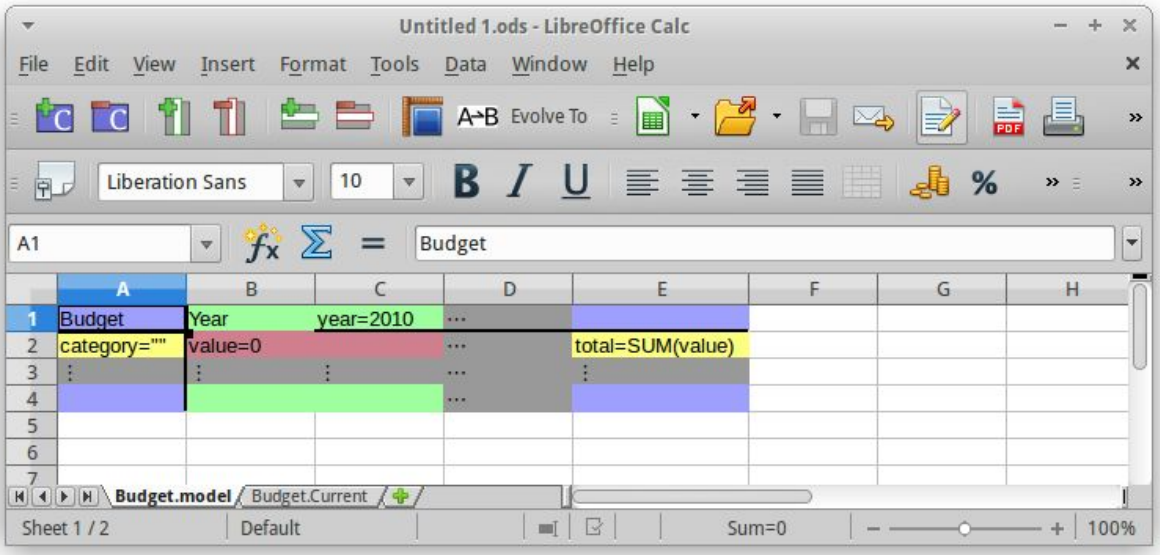
Inner classes can overlap if the previous rules are met.



When two classes overlap, a new class is added where both intersect, removing any ambiguities.

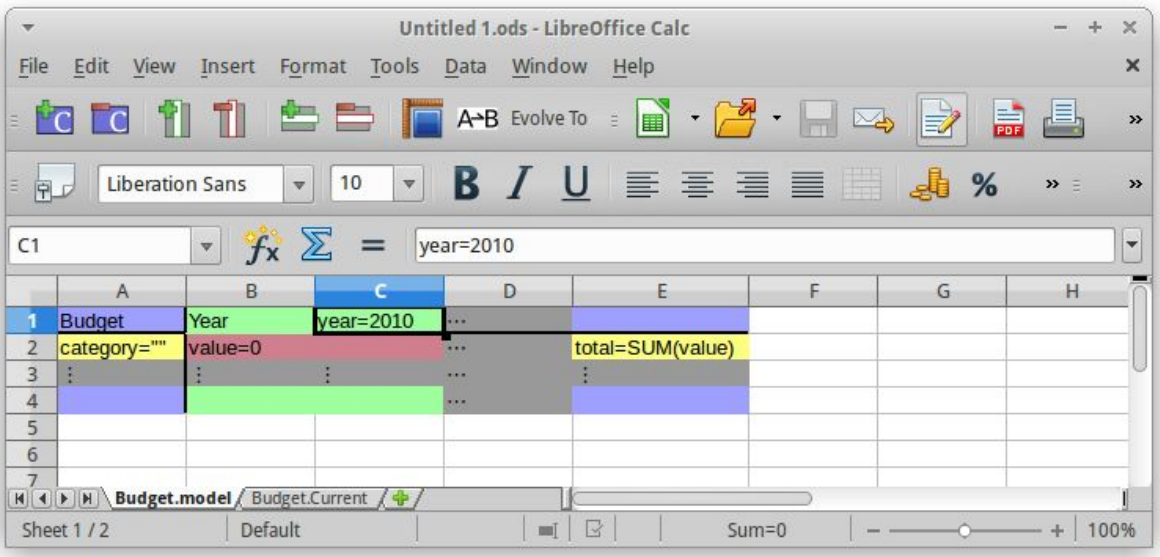


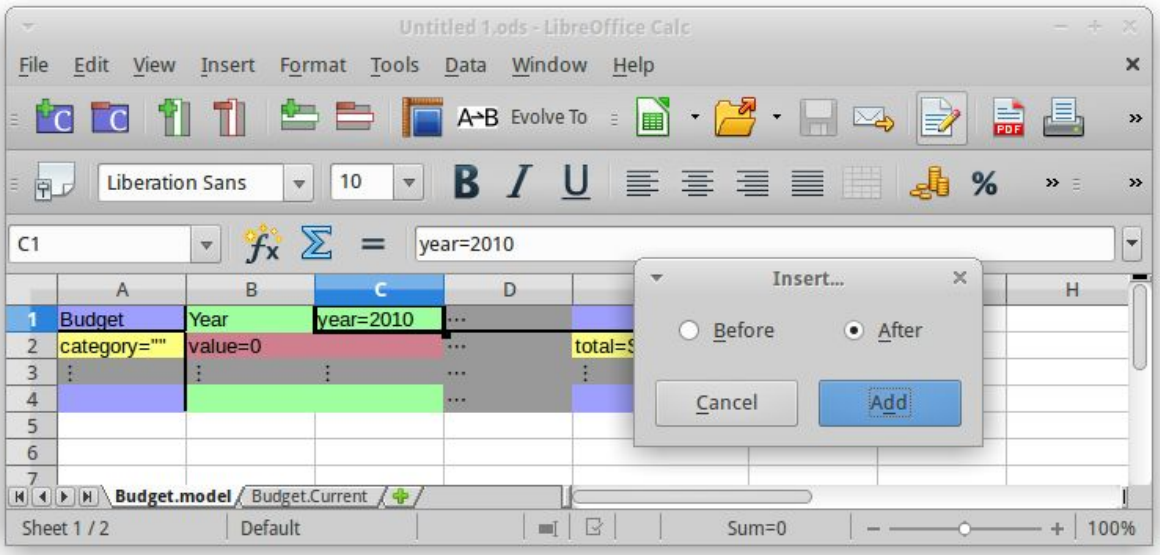
Cells in the data can be specified by simply writing in the corresponding cell in the model.

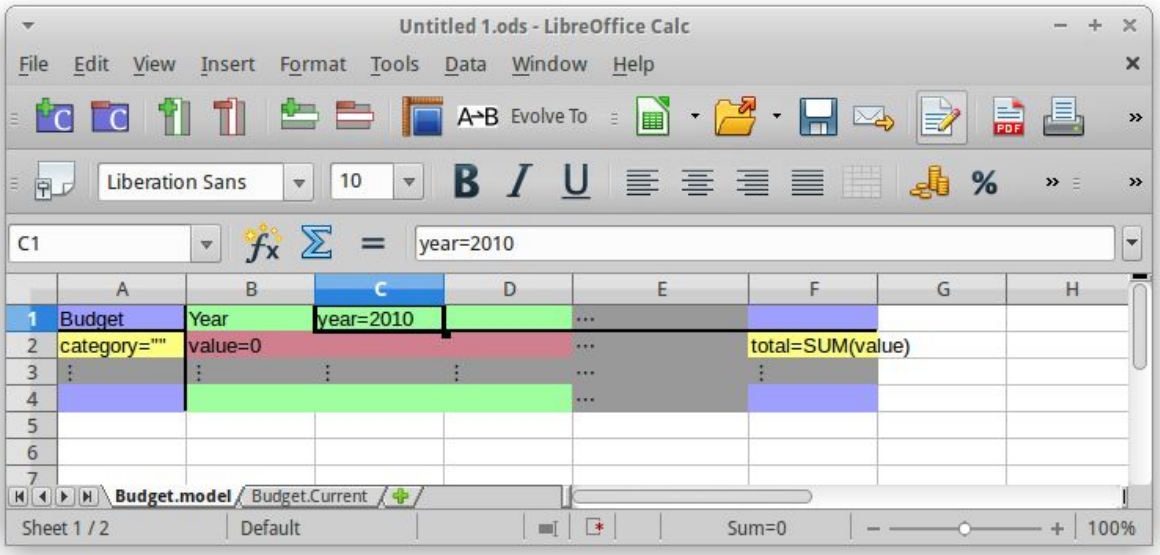


Model Evolution

Models can be evolved by adding columns (), removing columns (), adding rows (), and removing row (). For that, a cell must be selected, and then just press the button for the desired action. The images below show the steps to add a new column after column C.


Untitled 1.ods - LibreOffice Calc
File Edit View Insert Format Tools Data Window Help
Evolve To
Liberation Sans
10
C1
year=2010
Budget
Year
year=2010
category=""
value=0
total=SUM(value)
Budget.model
Budget.Current
Sheet 1 / 2
Default
Sum=0
100%


Untitled 1.ods - LibreOffice Calc
File Edit View Insert Format Tools Data Window Help
Evolve To
Liberation Sans
10
C1
year=2010
Insert...
Before
After
Cancel
Add
Budget
Year
year=2010
category=""
value=0
Budget.model
Budget.Current
Sheet 1 / 2
Default
Sum=0
100%


Untitled 1.ods - LibreOffice Calc
File Edit View Insert Format Tools Data Window Help
Evolve To
Liberation Sans
10
C1
year=2010
Budget
Year
year=2010
category=""
value=0
total=SUM(value)
Budget.model
Budget.Current
Sheet 1 / 2
Default
Sum=0
100%