Creating Configuration Extensions

Writing LibreOffice configuration extensions is the best method of managing a customized set of default settings for your organization. Simply install the shared extension along with LibreOffice, and all users on that computer will use those settings as default options.

Advantages of configuration extensions:

- approved and supported modification vector.
- extension settings override (have precedence over) system settings, but are overridden by the user's profile settings. So, you can easily override the LibreOffice install defaults, and still allow the user to set their own preferences.
- settings *can* be enforced if necessary, preventing users from altering those particular settings.
- affects both new users *and* users with existing profiles (existing profiles affected if the settings were *not yet* defined in the user's profile and also by enforced settings).

Disadvantages of configuration extensions:

- not centrally managed - each computer needs to be upgraded individually if changes to the extension are made. (Unless you register your extension with LibreOffice Extensions?)
- difficult to create the extension because it can be hard to isolate the XML needed to identify the configuration setting. Debugging a non-working extension is also difficult. Hopefully this document helps to overcome this disadvantage.

A good sysadmin will try to make the user experience as simple as possible, and that involves identifying default settings that are not ideal for his organization's particular situation. Some common examples of LibreOffice default settings to change include:

- save as .doc, .xls
- disable automatic, online upgrades
- setting the locale to ensure correct paper sizes like A4.
- setting default fonts

In this document, we will step through the process of how to create a configuration extension, by implementing the common optimization suggestion of setting the Graphics cache to be 100MB instead of the default value of 20MB. A sample extension is also provided that contains this setting and all of the common settings mentioned above.

You need to know a bit about XML in order to create a configuration extension, and this document assumes that you know enough. XML files can be edited with a normal text editor. We will be working primarily with the XML file <USER SETTINGS>/registrymodifications.xcu.

- Windows: <USER SETTINGS> = %APPDATA%\libreoffice\4\user\
- Linux: <USER SETTINGS> = ~/.config/libreoffice/4/user/

Step 1: Identifying the configuration parameter

a. Identify the LibreOffice location of the setting you want to change (because we want to be able to go directly to that location without hunting for it in later steps) and then close LibreOffice completely.
   - for our example: Tools Menu→ Options → LibreOffice → Memory

b. With LibreOffice closed, delete <USER SETTINGS>/registrymodifications.xcu. (This is an XML file that contains the user-specified settings. I am suggesting that you delete it so that it doesn't contain old settings and so that you will be working with a smaller file – making it much easier to locate the setting that you are attempting to change.)

c. Open Libreoffice, and directly go to the settings page, *but do not change any settings*. Then close LibreOffice completely, and backup the newly created <USER SETTINGS>/registrymodifications.xcu. (Some settings are written to the <USER SETTINGS> just by *visiting* the configuration page. Now we have a minimal baseline registrymodifications.xcu that we will use to compare to the one created in the next step.)
   - in this example, copied as registrymodifications.orig

d. Open LibreOffice and directly go to the settings page. Change the setting to the desired value. Then close LibreOffice completely and backup <USER SETTINGS>/registrymodifications.xcu which now contains your setting change.
   - In this example, copied as registrymodifications.memory

e. Isolate the differences between the two XML files. (This can be the hard part. Perhaps you can do it visually. I try to use xmldiff on Linux. Maybe there are other tools that do well at comparing XML files – however, most text file comparison tools don't work well with XML files. If you can't locate the change, repeat step 1.d, but set a different value. Then save as .memory2, and compare memory1 with memory2.)
   - in Linux: “xmldiff registrymodifications.orig registrymodifications.memory” (xmldiff can't handle very large files - another reason why it is good to start off by deleting and re-creating a minimal baseline setting.)
   - despite our care in trimming the .xcu file, a large number of changes are shown by xmldiff. Searching for “100” resulted in nothing. Searching for “graphic” or “cache” provides multiple lines, the most promising one (which actually is not the right one, but close enough) looks like this:

```
[update, /oor:items[1]/item[9]/@oor:path,
    /org.openoffice.Office.Common/Cache/GraphicManager]
[append-first, /oor:items[1]/item[9]/prop[1],
    <value>
        1048576
    </value>
```

f. Confirm that you have found the correct setting by editing the <USER SETTINGS>/registrymodifications.xcu XML file, and changing the value to something else. Then open LibreOffice and verify that the setting has changed.
   - Three entries exist for GraphicManager. I think the most likely one is TotalCacheSize
   - for our example, double the value to 209715200.
   - starting LibreOffice again shows that our cache limit indeed is now 200MB.

g. Isolate the full XML path of the setting.
   -

```
<items>
    <item oor:path="/org.openoffice.Office.Common/Cache/GraphicManager">
        <prop oor:name="TotalCacheSize" oor:op="fuse">
            <value>104857600</value>
        </prop>
    </item>
<items>
```

Step 2: Convert this XML into proper .xcu extension format

a. Configuration extensions properly use .xcu format.
   - https://wiki.openoffice.org/wiki/Documentation/DevGuide/Config/Customizing_Configuration_Data
b. Convert the first section of <item oor:path> to <oor:component-data> format. Make <node> elements out of the rest of oor:path. Adding oor:finalized allows you to ENFORCE this setting - preventing the user from over-riding it with his own preference. (Sometimes the finalized settings are visibly disabled in the GUI. In some other cases, if the user tries to modify it, LibreOffice will crash/close with the error "configmgr modification of finalized item".)
   - <oor:component-data oor:name="Common" oor:package="org.openoffice.Office"
     xmlns:oor="http://openoffice.org/2001/registry"
     xmlns:xs="http://www.w3.org/2001/XMLSchema" >
     <node oor:name="Cache">
     <node oor:name="GraphicManager">
     <prop oor:name="TotalCacheSize" oor:type="xs:int" oor:finalized="false">
     <value>104857600</value>
     </prop>
     </node>
     </node>
     </oor:component-data>
c. Is oor:type important? Can it be left out? Or leave it as "fuse" like it is in registrymodifications.xcu? Valid types include string, boolean, int, short, long, double, binary

Step 3: Package your custom .xcu files into a template

a. Create a directory to store your template in. Make a subdirectory "META-INF" as well.
   - in this example: lo_config_extension/META-INF
b. Group all settings with the same <oor:component-data oor:name> together into a single file. Separate files should be used for different sections of the registry.
   - in this example: lo_config_extension/common.xcu
c. Create a manifest.xml file in META-INF with a list of all of the config files you are adding.
   - <?xml version="1.0" encoding="UTF-8"?>
     <manifest:manifest>
     <manifest:file-entry manifest:full-path="common.xcu"
     manifest:media-type="application/vnd.sun.star.configuration-data"/>
     </manifest:manifest>
d. Compress the contents of the template folder into a .zip file and rename as .oxt
   - zip -r my_config_extension.oxt "."

Step 4: Debugging your package

a. Install as a shared extension from a command line. (This helps to see any error messages related to the extension or XML parsing.)
   - add with: unopkg add –shared my_config_extension.oxt
   - remove with: unopkg remove –shared my_config_extension.oxt
b. Mark each entry as oor:finalized=true. (This ensures that if the user has already set the value to something else, your value will show up anyway. In some cases you can tell that your setting has applied because if you try to change a finalized value Libreoffice will crash. Alternatively, if you don't use oor:finalized, you need to make sure that you first delete the test user's profile so you can see that the correct defaults are being applied.)
c. If you really run stuck, you could compile and run a debug version of LibreOffice that prints warnings to stderr.

Step 5: Deploy your package

a. Remember to reset oor:finalized=false for any settings that you don't want to enforce, especially any that cause a crash like TotalCacheSize.
b. Ensure that each .xcu file has an entry in manifest.xml
c. Add a description.xml file to make the extension feel polished, and provide useful features like version numbering and branding.
d. Add the installation of the extension into your LibreOffice installation routine. (Note that unopkg indicates that you should ensure that LibreOffice is *not* running when you add --shared extensions.)