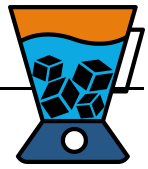
CuraBlender Documentation



This Plugin provides support for reading/writing BLEND files directly without importing/exporting.
It also offers extra features that are described in this document.

Table of contents

1. Explanation of all files

CuraBlender.py
The main module of this plugin. Contains the extension and general functions.
Those functions are:

- Loading and writing user settings.
- Setting and verifying the path to blender.
- Opening files in blender.
- File watcher for BLEND and foreign files for the 'Live Reload' function.

BLENDReader.py
The reader module of this plugin. Provides support for reading BLEND files.
Processes the file with the help of the BlenderAPI module. Counts the number of objects inside the file and reads them independently from each other. Gives files with multiple objects a special postfix with an index for reloading the correct object later.

BLENDWriter.py
The writer module of this plugin. Provides support for writing BLEND files.
Gets all files on the current build plate and saves them to a new BLEND file. Also works for split objects from BLEND files and foreign files (stl, obj, x3d, ply).

CuraBlender.qml
Builds the settings window for the extension. Contains all information about icons, buttons and checkboxes.

init.py
Registers the extension, the mesh reader and the mesh writer inside cura on startup.

BlenderAPI.py
The interface module between cura and blender. Uses the blender python API to work with blender objects. Contains four different program modes:

- **Count nodes:** Gets called everytime before reading loading the actual objects. Counts the number of objects inside the file and decides which mode to use.
- **Single node:** Gets called when file only contains one object. Removes decorators and loads the object.
- **Multiple nodes:** Gets called when file contains multiple objects. Removes decorators and loads the object based on given index. This program gets called for every object inside the file.
- **Write prepare:** Gets called right before the write step. Prepares the scene in blender.
- **Write:** Gets called on writing to a blender file. Loads objects from BLEND files based on index and imports foreign files.

plugin.json
Contains some information about the plugin.

images
Icons used by the settings window and for this documentation.

.gitignore
A file used by git to ignore selected files on commiting.

Documentation
Contains documentation, manual, hero shot and test review for this plugin.

LICENSE
A license file.

2. Potential problems

Ultimaker Cura Version
This plugin was developed and successfully tested on versions for **Ultimaker Cura from 4.0 up to 4.8**.
It should also work for future versions, but is not guaranteed.

Blender Version
Because this plugin closely works with Blender and it's python API, there could occur problems with new Blender updates. Nearly every update of blender changes commands of it's python API and could potentially need some tweaks.
This is especially true between blender 2.7x and blender 2.8x where the concept of collections was introduced. Therefore this plugin doesn't work for blender version 2.79 and below.
Successfully tested on **blender 2.80, blender 2.81, blender 2.82, blender 2.83, blender 2.90, blender 2.91**.

Default plugins inside Cura
This plugin reads files by converting them into another file type. This plugin supports (stl, obj, x3d, ply). The readers for those files are implemented in cura by default. However some readers are implemented as plugins. All are installed in cura by default, but could potentially be missing.

File conversion
BLEND files are not manually processed, but converted into other file types `see: Design decisions`.
A temporary file gets created in the same directory as the original file. This could lead to several problems. The path could be protected or the user doesn't have any write permission. Also if the original file gets deleted during a process, this could lead to further problems.

File explorer
If it's the first use time and and the plugin cannot find the blender path automatically, a file explorer window will open up.
However opening a file by the folder icon on the top right corner or by pressing `Open file(s)` will open another file explorer as if dragging a BLEND file into the cura window.
If the user drags a BLEND file into the cura window and cancels this process, cura might crash.
The user can set the blender path manually by calling the Debug Blenderpath function inside the extensions tab.

3. Design decisions

In this section several design decisions will be further explained.

Structure:
The main python modules of this plugin are split by category. This was done intentionally to enhance the readability.
`CuraBlender.py` contains all general functions and is used as an interface between the other modules.
`BLENDReader.py` contains methods to read a BLEND file.
`BLENDReader.py` contains methods to write a BLEND file.
`BlenderAPI.py` contains everything used from the blender library.

File conversion vs. manual processing:
Reading/Writing a BLEND file is not done by manually processing the mesh data, but by converting it to a prechosen file type.
Processing the mesh data manually wouldn't make much sense, because it is very error-prone.
Therefore this plugin uses blender to convert BLEND files to the desired file type and vice versa.
Every process is an instance of blender inside a `silent` shell. This shell does everything in background without the user noticing it.
The temporary file is created in the same directory as the original file. This is done for simplicity reasons. Another option would be to save those temporary files in any other directory like %appdata%.

Node indexing:
When processing BLEND files with multiple nodes, the plugin pre-processes the file with a counting method. The number of nodes is returned in a PIPE by subprocess.
Every object of this file gets a special postfix and is also indexed. Then every object is processed separately and mostly in parallel to increase load times.
The postfix with the index is important for reloading and writing the objects with the original file. The implementation needs to be uniformly everywhere the index is used. Otherwise the wrong object will be reloaded or written.

Reloading:
When reloading files in cura, it isn't possible to add or reduce the number of nodes on the current build plate. This means if the user opens a file in blender and adds or removes an object there, no object on the current build plate in cura will be added or removed.
This is the case because cura doesn't reload a file, but the node related to the file. The plugins implementation sticks closely to this concept. It's only possible to change and (live) reload nodes on the current build plate.

Logs:
This plugin creates some exception logs. These exceptions do not exceed the frame and are reduced to a minimum.

Preferences:
This plugin saves all settings with the preferences module from uranium.

- **Blender path:** The blender path inside this settings file gets set automatically or by the user with the help of the file explorer when the plugin cannot find it.
- **Import Type:** Blender files get converted into another file type on reading/writing (stl, obj, x3d, ply).
- **Live Reload:** Automatically reloads the object when changed in blender.
- **Auto Arrange on Reload:** Auto arranges the complete build plate after 'Live-Reload'.
- **Auto Scale on Read:** Scales the object down/up automatically to fit the build plate.
- **Show Scale Message:** Shows or hides the auto scale message.
- **Warn before closing other Blender instances (Caution!):** Shows or hides the message for closing other blender instances when opening a new one. Potential loss of data. Deactivate on own risk.

Verifying blender path and version:
The path to blender gets verified everytime something is processed by this plugin.
Also the version of blender is being checked for compatibility (blender version 2.80 or higher is required).
Although this consumes some time, it is done to prevent a wrongly set path by the user or automatically by this plugin.

Foreign files in blender:
If a foreign file (stl, obj, x3d, ply) gets opened in blender through the this plugin, a BLEND file for it is being created. This file stays and is not being removed by the plugin, because a file watcher is added to this file and if the file gets removed, the file watcher will not work anymore.

Time measurement:
All methods were tested and optimized with pythons time module. Of course the loading times could be increased for the cost of security and validating.
To measure the time of a specific section simply use `start = time.time()` before the specific section and write `time.time() - start` in the log file after the specific section.

4. Platform Support

This plugin works on every platform (**Windows**, **MacOS**, **Linux**) with full functionality.
To achieve this, a lot of platform specific fixes were made.
Some examples:

- Subprocess uses a list of arguments or a tuple of arguments as one big argument on windows, while on macOS and linux only the first argument gets executed. Therefore all commands are one big string.
- File watcher on windows crashes when a file from a removable device (usb, ...) gets opened. To fix this, a QEventLoop gets created before adding a path to the file watcher. Later this QEventLoop would be created anyways.
- Closing other blender instances is done with different commands.

5. Known challenges

- Adding/Removing objects inside a loaded BLEND file, doesn't add/remove the corresponding object in cura.
  This plugin links objects from blender to cura by index. Adding/Removing an object would mess this up.
  One idea is to hand over the object names from blender and add this to the file reference (file name).
- Grouped objects cannot be opened due to file reference reasons. This could be handled by saving all grouped objects into a new blend file and open the new file.
- When writing a blend file every object from a referenced file gets written to the new file, even if the user removed it from the build plate.