Simple G-Code Generator Version 3.6.0

USER MANUAL

Version 170725

Content

Introduction	2
Copyright	
License	.2
System requirements	
nstallation	
Quick start guide	
Specific default Pre- and Postamble	.4
Tables for feeds and speeds	
How to add milling features	.4

Introduction

The purpose of this Python application is to create g-code for cnc milling/routing machines for hobbyist use.

Copyright

Copyright (C) 2017 Erik Schuster erik at muenchen - ist - toll dot de

License

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see http://www.gnu.org/licenses/.

System requirements

The application was tested with the following systems:

- i7-4712MQ 8GB, Ubuntu Linux 16.04 (precise) 4.4.0-72-generic (64-Bit),
 Python 2.7.12, LinuxCNC 2.8.0~pre1 Simulation
- Atom330 1GB, LinuxCNC-2.7 wheezy, Python 2.7.12, LinuxCNC 2.7.8

Installation

Extract the package and move the Folder <SimpleGcodeGenerator> to the destination where you want it.

Make the Python file <SimpleGcodeGenerator vX.X.X.py> executable.

Edit the paths in the section <LINUXCNC> of the ini-file <sgg.ini>.

Edit the default values in the ini-file <defaults.ini> to your needs.

CXF-Fonts for text engraving

Since I am not sure about the license, I do not provide the fonts which come with the QCAD package.

A package can be downloaded here:

http://forum.librecad.org/attachment/4655223/0/FONTS.zip

Quick start guide

- Select "Preamble" in the listbox
 Created Objects>.
- <Double-Left-Click> on "OutlineCircle" in the listbox <Create Object>

The parameter edit frame <OutlineRectangle> will show up and the new object will be inserted in the <Created Objects> list after "Preamble".



Width

Height

Contour

Bridges

Shape rotation x

Shape rotation y

Shape rotation

Bridges width

Bridges height

Machining

direction

20

inside

outside

0

0

V

1.0

1.0

conventional

@ climb

exact

OutlineRectangle

Name of object OutlineRectangle 1

1.0

25.0

2.5

1500.0

1500.0

1500.0

30000.0

5.0

-5.0

0.0

0.0

0.0

0.0

Tool Number

Tool Diameter

Step over xy

Increment z

Feedrate z

Spindle speed

Safety height

Start height

End height

Position x

Position y

Rotate X

Rotate Y

Rotation XY

Feed rate xy @td

Feedrate xy @so

- Now the parameters of this object can be edited. The changes take effect almost immediately. UNDO is not possible.
- <Right-Click> on the entry fields of <Tool Number> or <Tool Diameter> will show up the tool selection frame.
- <Right-Click> on the entry fields of <Feed rate xy @td>,
 <Feed rate xy @so> and
 <Spindle speed> will show up the feeds and speeds frame.
- <Left-Click> on <Set pre- and postamble
 postamble> will show up the object specific options for the pre- and postamble.

Specific default Pre- and Postamble

If you need your own specific pre- and postambles, you have to choices:

- Edit the source code (not recommended)
- Create two ngc-files namend reamble.ngc> and <postamble.ngc>, placed
 in the <SimpleGcodeGenerator> folder. They will be loaded at start-up.

Tables for feeds and speeds

For adding more tables check out the file <Tables.ods> in the folder "data/feedsnspeeds". Create a new sheet from an existing one and export it as csv.

The programm expects csv-files with semicolon sepertated data and the data beginning with one line of headers after the keyword "Data" in the line before.

How to add milling features

To add a feature (eg. Pocket milling) the source files "ncclasses.py" and "guiclasses.py" must be edited.

Find a short howto below. Please contact the author for detailed questions if neccessary.

1. Add the logic

- Source file: "ncclasses.py"
- If you want to use all the basic parameters from "class Basedata(object)" copy the class "class TEMPLATE(Basedata, Basemethods)" and mofify to your needs. The available g-code classes are defined in the file "gcode.py".
- If you do NOT want the basic parameters, copy the class "Custom code"and modify it to your needs.
- Add the name of the new class to the list "NCCLASSES" at the end of the file.

2. Add the gui

- Source file: "guiclasses.py"
- If you want to use all the basic parameters from "class Basedata(object)" copy the class "class TEMPLATE(Basedata, Basemethods)" and mofify to your needs.
- If you do NOT want the basic parameters, copy the class "Custom code"and modify it to your needs.

- Add the name of the new class to the list "GUICLASSES" at the end of the file.
- 3. Check implementation.
 - Start the application.
 - Check if the new class appears in the CreateObject-List.
 - Create a new object with the new class and start bugfixing ;-)
- 4. Send the new classes to the author if you want.