
stl2obj

Release 0.3

Neizvestnyj

Oct 18, 2022

CONTENTS

1	Best and fastest python 3D files converter based on c++ functions	1
1.1	Usage	1
1.2	Examples	2
1.3	Installation	6

BEST AND FASTEST PYTHON 3D FILES CONVERTER BASED ON C++ FUNCTIONS

1.1 Usage

1.1.1 Stl2Obj - root class

Base convert files example

```
from stl2obj import Stl2Obj

src = 'path-to-src-file' # .stl or .obj
dst = 'path-to-dst-file' # .stl or .obj

Stl2Obj().convert(src, dst)
```

convert function takes:

- src - str path to .stl or .obj file
- dst - str path to output .obj or .stl file
- debug - bool disable c++ cout
- callback - object callback function in python, default None
- progress_callback - object called when part of the file processing has been completed, it is always 1, so do in your project

```
def progress_callback(self, value: int):
    self.total_progress += value
```

stl_mode_converter function takes:

- src - str path to .stl file
- dst - str path to output .stl file
- mode - str ASCII, BIN or AUTO
- callback - object callback function in python, default None
- progress_callback - object like in convert function

1.2 Examples

1.2.1 Convert stl to obj

```
from stl2obj import Stl2Obj
import os

files_folder = 'files'

src = os.path.join(files_folder, 'cube.stl')
dst = os.path.join(files_folder, 'cube_converted.obj')

a = Stl2Obj()
a.convert(src, dst)
```

1.2.2 Convert obj to stl

```
from stl2obj import Stl2Obj
import os

files_folder = 'files'

src = os.path.join(files_folder, 'pumpkin.obj')
dst = os.path.join(files_folder, 'files', 'pumpkin_converted.stl')

a = Stl2Obj()
a.convert(src, dst)
```

1.2.3 Kivy example

```
from kivy.app import App
from kivy.lang.builder import Builder
from kivy.uix.modalview import ModalView
from kivy.logger import Logger
from kivy.graphics.vertex_instructions import GraphicException

from render import Renderer

from stl2obj import Stl2Obj
import pathlib
import os
import time
import threading

KV = """
BoxLayout:
    ScrollView:
        BoxLayout:
            id: box
```

(continues on next page)

(continued from previous page)

```

orientation: 'vertical'
size_hint_y: None
height: self.minimum_height
spacing: dp(20)
padding: dp(5), dp(20), dp(5), dp(20)

TextInput:
    id: field_stl
    text: 'Dolphin.stl'
    size_hint_y: None
    height: self.minimum_height

TextInput:
    id: field_obj
    text: 'Dolphin.obj'
    size_hint_y: None
    height: self.minimum_height

ProgressBar:
    id: pb

Label:
    id: log_lbl
    text: ''
    size_hint_y: None
    height: self.texture_size[1]

Widget:
    size_hint_y: None
    height: max(root.height - dp(50 * 3) - field_obj.height * 2 - dp(20 * 7) -
↳ dp(20 * 2) - log_lbl.height, dp(10))

Button:
    id: run
    text: 'Run'
    size_hint_y: None
    height: dp(50)
    on_release: app.convert()

Button:
    id: convert
    text: 'Convert stl mode'
    size_hint_y: None
    height: dp(50)
    on_release: app.convert_mode()

Button:
    text: 'Preview'
    id: preview
    size_hint_y: None
    height: dp(50)
    on_release: app.preview()

```

(continues on next page)

(continued from previous page)

```

"""

def run_in_thread(fn):
    def run(*k, **kw):
        t = threading.Thread(target=fn, args=k, kwargs=kw)
        t.daemon = True
        t.start()
        return t

    return run

class TestApp(App):
    def __init__(self, **kwargs):
        super().__init__(**kwargs)
        self.render = None
        self.view = None

    def build(self):
        return Builder.load_string(KV)

    def on_start(self):
        self.get_files_folder()

    @staticmethod
    def get_file(file: str):
        files_folder = os.path.join(pathlib.Path(__file__).parent.resolve(), 'files')
        return os.path.join(files_folder, file)

    @run_in_thread
    def convert(self):
        start_time = time.time()
        self.reset_widgets(True)

        src = self.get_file(self.root.ids.field_stl.text)
        dst = self.get_file(self.root.ids.field_obj.text)

        try:
            Stl2Obj().convert(src=src,
                             dst=dst,
                             debug=True,
                             callback=self.callback,
                             progress_callback=self.progress)
            print(f'Conversion done behind {time.time() - start_time}')
        except (FileNotFoundError, TypeError) as e:
            self.reset_widgets(False)
            print(e)

    @run_in_thread
    def convert_mode(self):
        self.reset_widgets(True)

```

(continues on next page)

(continued from previous page)

```

    stl = self.get_file(self.root.ids.field_stl.text)
    output = stl.replace('.stl', '-converted.stl')

    try:
        f = open(stl, encoding='latin-1')
        first_line = f.readline()

        if 'solid' in first_line:
            # like `get_stl_mode` in c++
            mode = 'ASCII'
        else:
            mode = 'BIN'

        self.root.ids.log_lbl.text = f'.stl file in {mode} mode'

        Stl2Obj().stl_mode_converter(src=stl, dst=output, callback=self.callback,
↪progress_callback=self.progress)
        print('Conversion mode done')
    except (FileNotFoundError, TypeError) as e:
        self.reset_widgets(False)
        print(e)

    def reset_widgets(self, reset: bool):
        self.root.ids.run.disabled = reset
        self.root.ids.convert.disabled = reset
        self.root.ids.preview.disabled = reset
        self.root.ids.log_lbl.text = ''
        self.root.ids.pb.value = int(reset)

    def callback(self, code: int):
        print(f'Callback code {code}')
        self.reset_widgets(False)

        if code == -1:
            raise TypeError

    def progress(self, value: int):
        self.root.ids.pb.value += value

    def preview(self):
        try:
            if not self.render:
                print(self.root.ids.obj)
                obj = self.get_file(self.root.ids.field_obj.text)
                self.render = Renderer(source=obj)
                self.view = ModalView(size_hint=(0.8, 0.8))
                self.view.add_widget(self.render)

            self.view.open()
        except (OverflowError, GraphicException) as e:
            # https://github.com/kivy/kivy/issues/7105#issuecomment-1195393114

```

(continues on next page)

(continued from previous page)

```
        Logger.critical(msg=f'{e}')

def get_files_folder(self):
    """
    :return:
    If you want to use example immediately after downloading package
    """

    files_path = os.path.join(pathlib.Path(__file__).parents[2].resolve(), 'files')

    if os.path.exists(files_path):
        if any('.stl' in file for file in os.listdir(files_path)):
            self.root.ids.field_stl.text = os.path.join(files_path, self.root.ids.
↪field_stl.text)
            self.root.ids.field_obj.text = os.path.join(files_path, self.root.ids.
↪field_obj.text)

TestApp().run()
```

1.3 Installation

1.3.1 Python

Pypi: `pip install stl2obj`

GitHub: `pip install https://github.com/Neizvestnyj/stl-to-obj/master.zip`