

## Instructions on Running the Code

The code for this project was built as an independent entity, and can be run via Terminal with UI command line.

The interface offers plenty of flexibility and automation.

It has its own interface, with default values and helper functions with a built-in manual on how to operate.

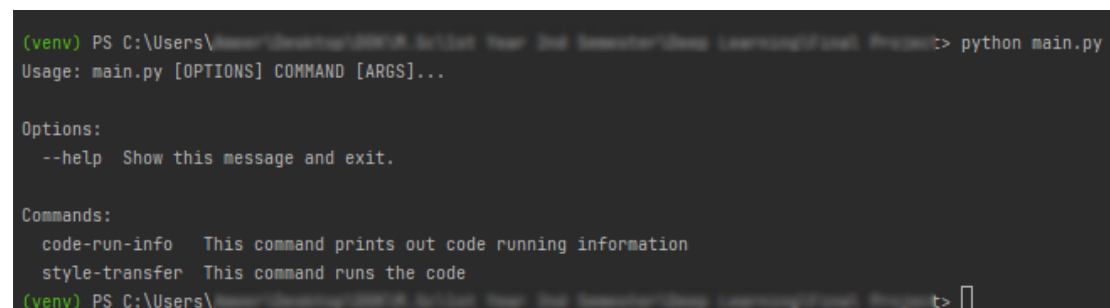
The Command Line Interface was built using Click Module Library, and designed in such a way that could be run on an ambient directory (and all of its content). Further, the code is designed to have the ability to run on multiple images (as specified in the CLI code running parameters command). If such were to be chosen, the code selects  $N$  (as specified) random Content and Style images for Style Transfer combinations.

## Application

If to be seeking and exploring the CLI,

Let's run, and hence illustrate, some commands and walk through the process.

Running *python main.py*:

A terminal window with a dark background and green text. The prompt is '(venv) PS C:\Users\...'. The command entered is 'python main.py'. The output shows the usage 'main.py [OPTIONS] COMMAND [ARGS]...' and lists two options: '--help Show this message and exit.' and two commands: 'code-run-info This command prints out code running information' and 'style-transfer This command runs the code'. The prompt returns to '(venv) PS C:\Users\...' with a cursor.

```
(venv) PS C:\Users\...> python main.py
Usage: main.py [OPTIONS] COMMAND [ARGS]...

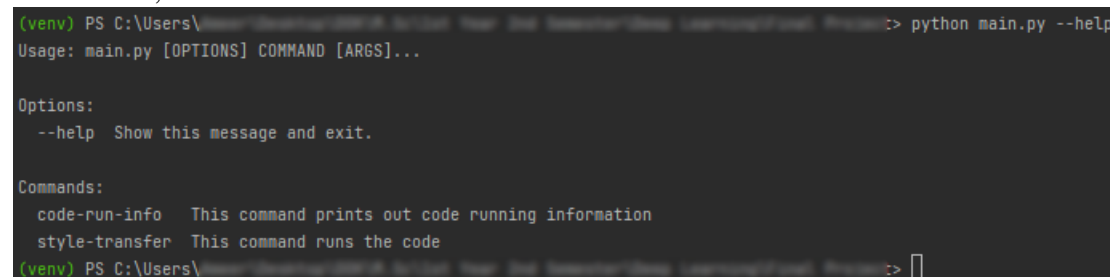
Options:
  --help  Show this message and exit.

Commands:
  code-run-info  This command prints out code running information
  style-transfer  This command runs the code
(venv) PS C:\Users\...>
```

Figure 1: Running *python main.py* command

We can notice, when not specifying any parameters, the CLI doesn't run the code, but instead print some information on the used built-in functions the user may use, and indicates for a `--help` command which could be applied.

If to be ran,

A terminal window with a dark background and green text. The prompt is '(venv) PS C:\Users\...'. The command entered is 'python main.py --help'. The output is identical to Figure 1, showing the usage, options, and commands. The prompt returns to '(venv) PS C:\Users\...' with a cursor.

```
(venv) PS C:\Users\...> python main.py --help
Usage: main.py [OPTIONS] COMMAND [ARGS]...

Options:
  --help  Show this message and exit.

Commands:
  code-run-info  This command prints out code running information
  style-transfer  This command runs the code
(venv) PS C:\Users\...>
```

Figure 2: Running *python main.py --help*

Will out come in the same information message.

Hence, according to the description in the information message above, let's run the following,

```
(venv) PS C:\Users\...> python main.py code-run-info
In order to run the code, please choose a content directory path and style directory path in which the content and style images could be found

An example command for running the code:
python main.py style-transfer --content_dir_path=.content_dir --style_dir_path=.style_dir

Another example command is getting help:
python main.py
or
python main.py --help

For getting help on a specific function:
python main.py style-transfer --help
or
python main.py code-run-info --help
(venv) PS C:\Users\...>
```

Figure 3: Running *python main.py code – run – info*

```
(venv) PS C:\Users\...> python main.py code-run-info --help
Usage: main.py code-run-info [OPTIONS]

    This command prints out code running information

Options:
  --help  Show this message and exit.
(venv) PS C:\Users\...>
```

Figure 4: Running *python main.py code – run – info – –help*

We can read through the helping instructions and operate properly.

One command in particular which was printed is the following,

```
An example command for running the code:
python main.py style-transfer --content_dir_path=.content_dir --style_dir_path=.style_dir
```

Figure 5: A command via the helping section on *code run info* for how to run the code properly

Trying out the helper command for *style\_transfer* function (main core command for this code),

```
(venv) PS C:\Users\...> python main.py style-transfer --help
Usage: main.py style-transfer [OPTIONS]

    This command runs the code

Options:
  --content_dir_path TEXT      Content directory's path
  --style_dir_path TEXT       Style directory's path
  --number_content_style_images INTEGER
                               Number of Content and Style images which
                               will be randomly chosen to create a Target
                               final image
  --help                      Show this message and exit.
(venv) PS C:\Users\...>
```

Figure 6: Running *python main.py style – transfer – –help*

We have all the necessary information for running the code with the right parameters.

As such, if to run the code we can use multiple commands.

All relies on the fact that if any of the parameters wasn't to be specified (Yet, can't name it without an argument. If no argument to be used, then it's better to not mention the specific parameter), then a default value will be initiated (from the *config.py* file. Which its values in a *.env* file (a *.env\_template* is provided in the project directory)).

As an example,

If none of the parameters were chosen,

```
(venv) PS C:\Users\...> python main.py style-transfer

Warning: Taking content dir path from env variables

Warning: Taking style dir path from env variables

Content path taken: .\content_images
Style path taken: .\style_images
Number of Content and Style images which'll be randomly chosen is : 1

Running the Code...
(venv) PS C:\Users\...>
```

Figure 7: Running *python main.py style – transfer*

The code will always (up to a certain precision) output such a result. And, of course, the code starts running (as specified in the message at the bottom).

We can notice how the default arguments were applied.

Another variation for running the code with one of the 3 arguments,

```
(venv) PS C:\Users\...> python main.py style-transfer --number_content_style_images=2

Warning: Taking content dir path from env variables

Warning: Taking style dir path from env variables

Content path taken: .\content_images
Style path taken: .\style_images
Number of Content and Style images which'll be randomly chosen is : 2

Running the Code...
(venv) PS C:\Users\...>
```

Figure 8: Running *python main.py style – trasfer – --number\_content\_style\_images = 2*

One can notice how, now, the *number\_content\_style\_images* value was changed to 2 (rather than the default one above in figure 7).

That's that! Hope it's clear.

Have fun with the code, and try exploring out of the features!

**M. A**