

---

# Flask-Now Documentation

*Release*

Ozan Onur Tek

Feb 23, 2018



---

## Contents

---

<b>1</b>	<b>Overview</b>	<b>1</b>
<b>2</b>	<b>Goal</b>	<b>3</b>
<b>3</b>	<b>Quick-Start</b>	<b>5</b>
3.1	Requirements . . . . .	5
3.2	Installing(GNU/Linux - OSX) . . . . .	5
3.3	Usage . . . . .	5
<b>4</b>	<b>Parameter Passing</b>	<b>7</b>
<b>5</b>	<b>Supported Architectural Patterns</b>	<b>9</b>
5.1	simple . . . . .	9
5.2	mvc . . . . .	9
<b>6</b>	<b>Supported Flask-Extensions</b>	<b>11</b>
<b>7</b>	<b>Initial Content Of Files</b>	<b>13</b>
7.1	config.py . . . . .	13
7.2	run.py . . . . .	13
7.3	requirements.txt . . . . .	14
7.4	__init__.py . . . . .	14
7.5	controller.py . . . . .	14
7.6	models.py . . . . .	14
7.7	templates/index.html . . . . .	15
7.8	static/css/style.css . . . . .	15
7.9	static/js/script.js . . . . .	15
<b>8</b>	<b>Uninstalling</b>	<b>17</b>
<b>9</b>	<b>Detailed Example</b>	<b>19</b>



# CHAPTER 1

---

## Overview

---

Welcome to [flask-now 0.1.5](#) documentation. Flask-Now is a simple command line interface tool which can generate the architectural pattern, folder structures and flask extensions for your project.



## CHAPTER 2

---

### Goal

---

The aim of Flask-Now is auto-generating necessity folders and files according to your architectural pattern and semi-automatically installing desired [flask extensions](#).

So, just concentrate on application logic, Flask-Now is at your services :)





### 3.1 Requirements

- Python 3.x
- Virtualenvironment
- pip3

### 3.2 Installing(GNU/Linux - OSX)

Install flask-now with pip, this will add an executable file to your /bin with the name flask-now which is out application

```
pip install --user flask-now
```

### 3.3 Usage

Now go to your project folder, create your virtualenvironment, activate it and build your project with your favourite flask extensions and architectural pattern

```
flask-now -mvc wtf bootstrap sqlalchemy
```

That's it, your project is ready in a minute with desired flask extensions and with following mvc like architectural pattern,

```
/your_project_folder
  venv
  run.py
  requirements.txt
  /project
```

```
__init__.py
controller.py
models.py
config.py
/static
    /css
        style.css
    /js
        script.js
/templates
    index.html
```

which has a very similar folder structure as suggested in [official tutorial](#).

---

### Parameter Passing

---

Flask-Now accepts parameters within following structure:

```
flask-now -<architectural-pattern> <extension_1> <extension_2> ... <extension_n>
```

Which means that you can pass one parameter only as -<architectural-pattern>. If you pass more than one architectural-pattern parameters like in the following example:

```
flask-now -mvc -mvvm -simple
```

Flask-Now will use the latest parameter(in this case it will use `simple`) that you passed as architectural-pattern.

If no parameter is passed to Flask-Now, it will build a simple architecture with no extension:

```
flask-now
```

which will produce a Flask ready project with following folder structure:

```
/your_project_folder
  venv
  run.py
  config.py
  requirements.txt
```

It is a very simple Flask Application and of course, it is ready to run!

```
python3 run.py
```



---

## Supported Architectural Patterns

---

Flask-Now supports two type of patterns in this version, more patterns will be added in next releases.

### 5.1 simple

Running:

```
flask-now -simple
```

With `-simple` parameter, Flask-Now will generate very simple Flask Application with in following structure:

```
/your_project_folder
  venv
  run.py
  config.py
  requirements.txt
```

- If you don't pass any parameter as `architectural-pattern` Flask-Now will build a simple folder structure for your project:

```
flask-now
```

is the exact same thing with:

```
flask-now -simple
```

### 5.2 mvc

Running:

```
flask-now -mvc
```

With `-mvc` parameter, Flask-Now will generate very similar Flask Application which is suggested in [official tutorial](#).

```
/your_project_folder
  venv
  run.py
  requirements.txt
  /project
    __init__.py
    controller.py
    models.py
    config.py
    /static
      /css
        style.css
      /js
        script.js
    /templates
      index.html
```

---

### Supported Flask-Extensions

---

Flask-Now supports all [official extensions](#). You can build your Flask Application with extensions using following rule:

Generally, Flask Extensions are named as follows:

```
Flask-SQLAlchemy  
Flask-Themes  
Flask-WTF  
etc...
```

Which means that they have following pattern:

```
Flask-<extension-name>
```

except:

```
Frozen-Flask
```

So you just need to drop **Flask-** or **-Flask** keyword from the name of the extension. For example, if you wish to use Flask-WTF, Frozen-Flask, Flask-Static-Compress and Flask-SQLAlchemy in your project with mvc like architecture, you just need to run flask-now as follows:

```
flask-now -mvc frozen wtf static-compress sqlalchemy
```

That's it, Flask-Now will build your Flask Application with desired extensions and mvc like pattern! requirements.txt, config.py is also at your services!

You just have to do

```
python3 run.py
```





---

## Initial Content Of Files

---

### 7.1 config.py

Initially, content of `config.py` as follows:

```
DEBUG=True
SECRET_KEY="Ug1cHqJJhRrLHqwqXS56lKh4z977sHqbdJZF3Zdhknrv/
↪ato82t3RZ3nMwsy8Q3wN34ukRPYxhflq3e81gUgSw=="
SERVER_NAME="127.0.0.1:5000"
```

When Flask-Now generates `config.py`, it uses `os.urandom()` to generate `SECRET_KEY` as suggested in [flask quick start](#).

### 7.2 run.py

If it is built with `-simple` parameter, content of `run.py` as follows:

```
from flask import Flask

app = Flask(__name__)
app.config.from_pyfile("config.py")

@app.route("/")
def index():
    return "<h1>Hello World!</h1>"

if __name__ == "__main__":
    app.run()
```

if it is built with `-mvc` parameter, content of `run.py` as follows:

```
from project import app

if __name__ == "__main__":
    app.run()
```

## 7.3 requirements.txt

Flask-Now uses `pip freeze` feature to create latest version of `requirements.txt` file. So, if your run

```
flask-now -simple
```

Flask-Now will generate following `requirements.txt` for you:

```
click==6.7
Flask==0.12.2
itsdangerous==0.24
Jinja2==2.10
MarkupSafe==1.0
pkg-resources==0.0.0
Werkzeug==0.14.1
```

Following files are only available with mvc like architecture:

## 7.4 \_\_init\_\_.py

```
from flask import Flask, render_template

app = Flask(__name__)
app.config.from_pyfile("config.py")

from project import controller
```

## 7.5 controller.py

```
from project import app, render_template

@app.route("/")
def index():
    return render_template("index.html")
```

## 7.6 models.py

```
# Your models here.
```

## 7.7 templates/index.html

```
<h1>Flask is fun.</h1>
```

## 7.8 static/css/style.css

This file is initially empty.

## 7.9 static/js/script.js

This file is initially empty.



## CHAPTER 8

---

### Uninstalling

---

Deactivate your virtualenvironment, than:

```
pip uninstall flask-now
```



## CHAPTER 9

---

### Detailed Example

---

Assume that we want to start a new Flask Application called `flaskr` in current directory.

- Let's install Flask-Now first.

```
pip install --user flask-now
```

- After installation succeed, let's create our directory for project and go to that directory

```
mkdir flaskr && cd flaskr
```

- Let's create our virtualenvironment in flaskr directory

```
virtualenv -p python3 venv
```

- Activate our virtualenvironment

```
source venv/bin/activate
```

- Finally, build our project using Flask-Now

```
pip install -mvc -wtf -sqlalchemy -login
```

That's it, you are ready to develop your Flask Application!