

ATIVIDADE PRÁTICA

PRIMEIRO CONTATO COM O FLASK

Flask é um micro framework que utiliza a linguagem Python para criar aplicativos Web. Um micro framework são frameworks modularizados que possuem uma estrutura inicial muito mais simples quando comparado a um framework convencional. Ou seja, esse “micro” é na verdade uma versão minimalista de frameworks, sendo bastante utilizado para criação de microsserviços, como APIs RESTful.

O Flask depende de duas bibliotecas externas, Werkzeug e Jinja2. Werkzeug é um toolkit para WSGI, a interface padrão entre aplicações web Python e servidores HTTP para desenvolvimento e implantação. Enquanto o Jinja2 é voltado para renderização de templates.

A instalação é bem simples, bastando executar o comando:

pip install Flask

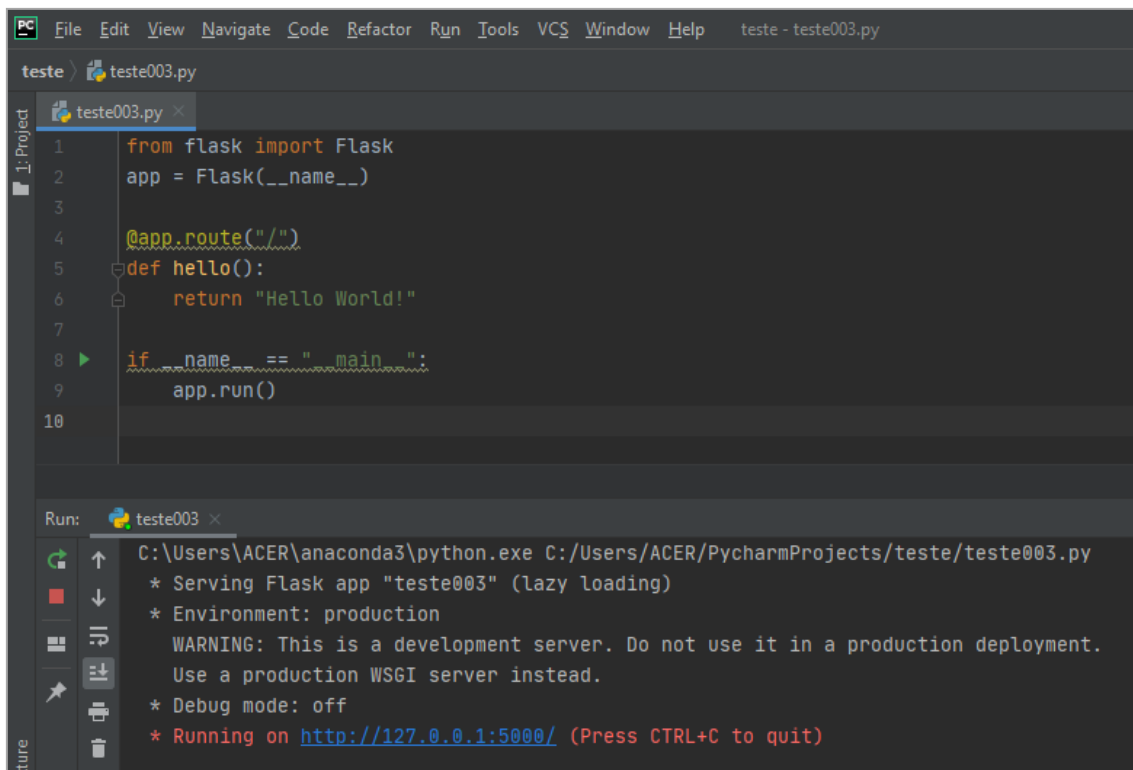
Quanto ao código de exemplo, temos um bastante básico:

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

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

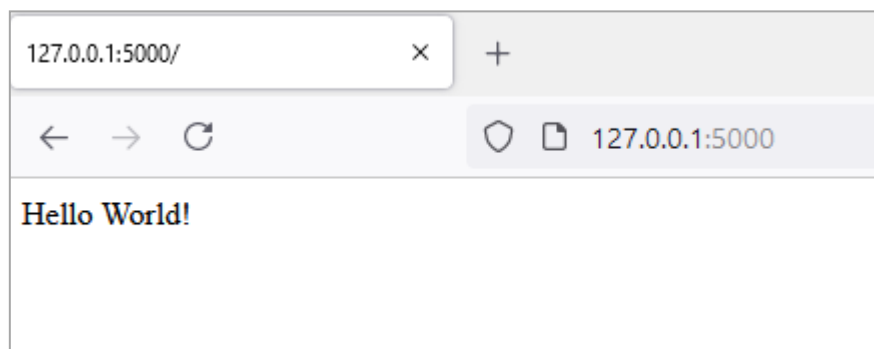
Ao executar o código, a mensagem de que existe um serviço rodando em “<http://127.0.0.1:5000/>” é apresentada. Isso significa que o código está “ouvindo” na porta 5000 da nossa própria máquina (localhost):



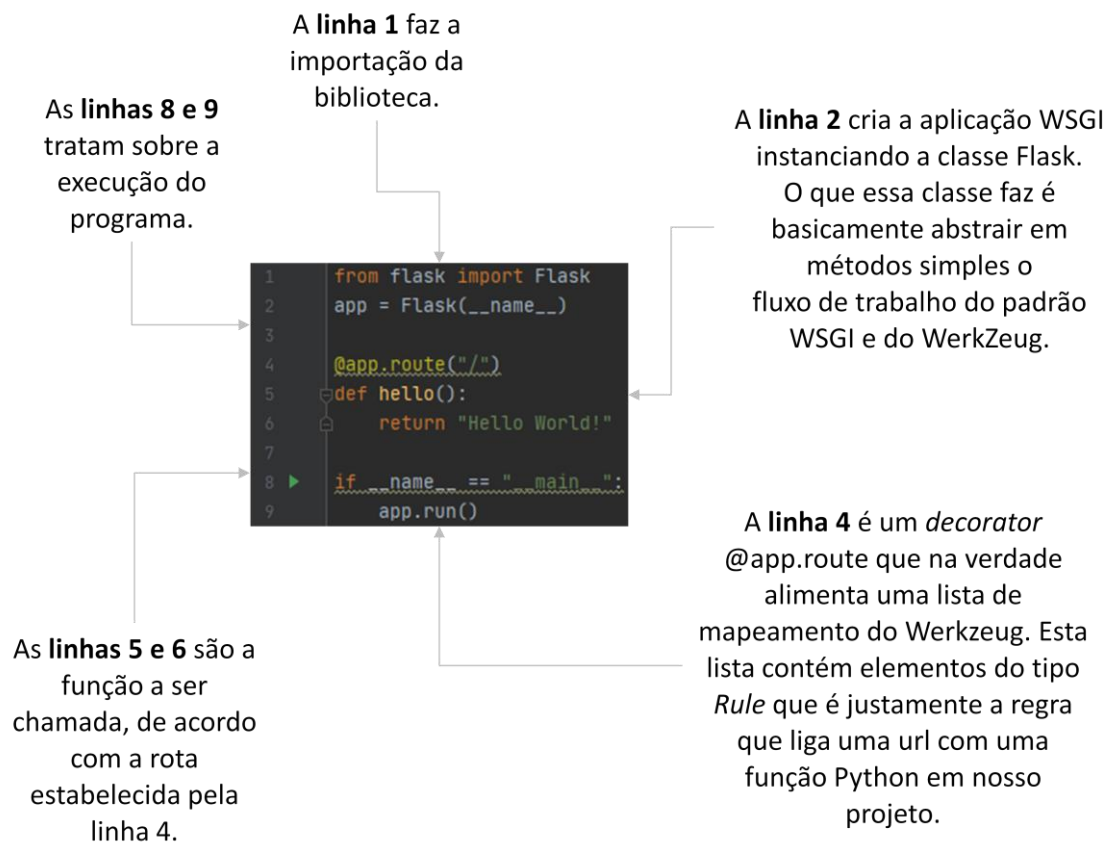
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help teste - teste003.py
teste > teste003.py
1 from flask import Flask
2 app = Flask(__name__)
3
4 @app.route("/")
5 def hello():
6     return "Hello World!"
7
8 if __name__ == "__main__":
9     app.run()
10

Run: teste003 x
C:\Users\ACER\anaconda3\python.exe C:/Users/ACER/PycharmProjects/teste/teste003.py
* Serving Flask app "teste003" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Para testar o programa abra um navegador (Firefox, Chrome, etc.), digite endereço no browser e observe a mensagem “Hello World” ser exibida. Isso significa que a aplicação Flask está executando com seu servidor web de testes (não se deve utilizar em aplicações de produção o servidor de testes disponível no Flask. Procure usar, por exemplo, o “Gunicorn”).



Explicação do código:



Crie um novo arquivo Python e teste o código abaixo. Entretanto, é preciso ter uma pasta chamada “templates” dentro de seu projeto com os arquivos “error.html” e “hello.html” para que funcione corretamente. Tais arquivos serão disponibilizados em aula.

Procure observar quais as rotas que o código de exemplo disponibiliza e os recursos que o Flask oferece! Veja a tela de exemplo com a saída do navegador, observe como o endpoint mudou.



```

from flask import Flask
from flask import render_template
from flask import jsonify
from flask import request

app = Flask(__name__)

#####
# RETORNANDO JSON

@app.route("/")
def index():
    return jsonify({"mensagem": "Hello Json!"})

#####
# UTILIZANDO METALINGUAGEM JINJA

@app.route('/hello/')
@app.route('/hello/<nome>')

def hello(nome=None):
    return render_template('hello.html', name=nome)

#####
# PASSAGEM DE PARÂMETRO - NÚMERO INTEIRO

@app.route('/show/<int:id>')
def show(id):
    return 'Valor recebido id = %d' %id

#####
# VERIFICANDO O METODO (VERBO HTTP) UTILIZADO

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        return 'Login via POST'
    else:
        return '''
        <html>
        <body>
        <form method="post">
            <p><input type="text" name="username">
            <p><input type="submit" value="Login">
        </form>
        </body>
        </html>
        '''

#####
# Pagina nao encontrada - erro 404

@app.errorhandler(404)
def page_not_found(error):
    return render_template('error.html'), 404

#####

if __name__ == '__main__':
    app.run()

```