Flask

Desenvolvendo aplicações Web com o Framework Flask - Aula 8

Flask-Bootstrap

Flask-Bootstrap

Na aula 4 desta seção vimos como adicionar os arquivos do *Bootstrap* manualmente em nosso projeto, nesta aula veremos como utilizar a extensão *Flask-Bootstrap*.

Podemos instalar esta extensão com o comando:

$ pip install flask-bootstrap

Flask

Flask-Bootstrap

As extensões *Flask* são inicializadas ao mesmo tempo que a instância da aplicação é criada. Veja como iniciar o *Flask-Bootstrap*.

```
from flask_bootstrap import Bootstrap
Bootstrap = Bootstrap(app)
```

Flask-Bootstrap

Depois que o *Flask-Bootstrap* é inicializado, um template-base, que inclui todos os arquivos do *Bootstrap* e a estrutura geral, torna-se disponível à aplicação.

A aplicação então tira proveito da herança de *templates* do *Jinja2* para estender esse template-base.

Flask

Flask-Bootstrap

Vamos criar um pequeno exemplo.

Arquivo app.py:

```
from flask import Flask, render_template
from flask_bootstrap import Bootstrap


app = Flask(__name__)
bootstrap = Bootstrap(app)

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

@app.route('/exemplo')
def exemplo():
    return render_template('exemplo.html', nome="Evaldo")

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

Flask-Bootstrap

Arquivo index.html:

```
{% extends "bootstrap/base.html" %}

{% block title %}Título{% endblock %}

{% block navbar %}

<div class="navbar navbar-inverse" role="navigation">
    <div class="container">
        <div class="navbar-collapse collapse">
            <ul class="nav navbar-nav">
                <li><a href="/">Principal</a></li>
                <li><a href="/exemplo">Exemplo</a></li>
            </ul>
        </div>
    </div>
</div>
{% endblock %}
...
```

```
...
{% block content %}
<div class="container">
    <div class="page-header">
        <h1>Esta é a página principal</h1>
    </div>
</div>
{% endblock %}
```

Flask

Flask-Bootstrap

Arquivo exemplo.html:

```
{% extends "bootstrap/base.html" %}
{% block title %}Título{% endblock %}
{% block navbar %}

<div class="navbar navbar-inverse" role="navigation">
    <div class="container">
        <div class="navbar-collapse collapse">
            <ul class="nav navbar-nav">
                <li><a href="/">Principal</a></li>
                <li><a href="/exemplo">Exemplo</a></li>
            </ul>
        </div>
    </div>
</div>
{% endblock %}
...
```

```
...
{% block content %}
<div class="container">
    <div class="page-header">
        <h1>Olá, {{ nome }}! Esta é a página de exemplo.</h1>
    </div>
</div>
{% endblock %}
```

Flask-Bootstrap

Os templates que criamos possuem três blocos, sendo *title*, *navbar* e *content*. São todos blocos exportados pelo *template-base* do *Flask-Bootstrap* para que nossos templates derivados possam definí-los.

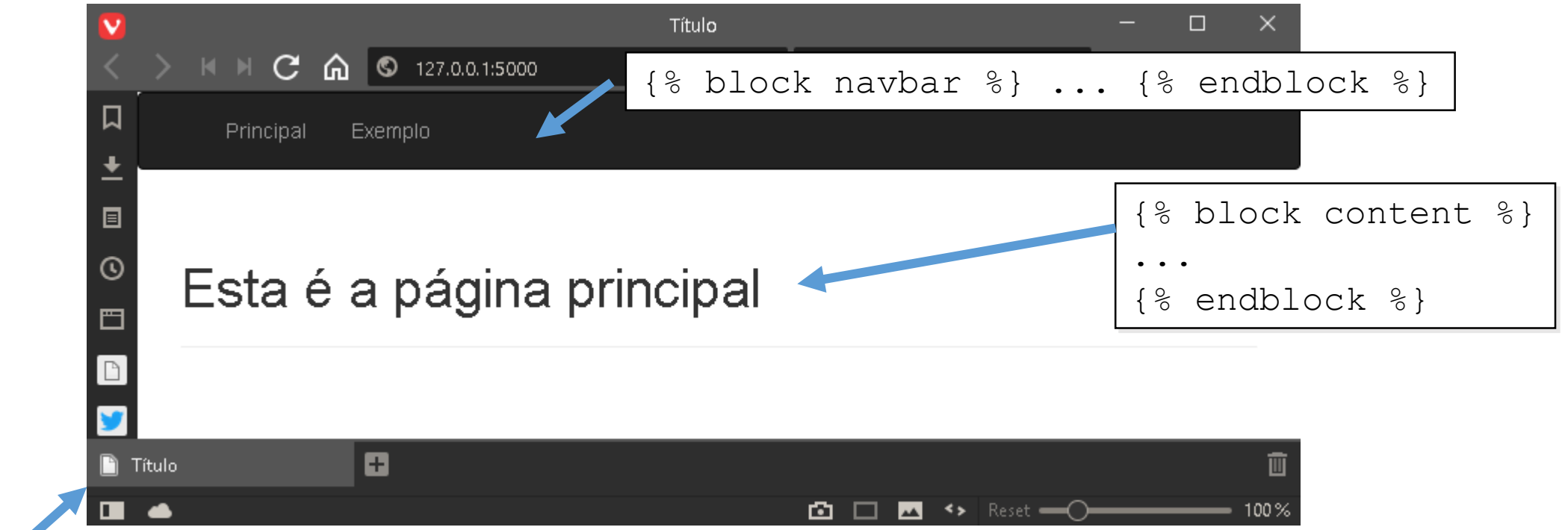
title: Define o conteúdo que aparecerá entre tags <title></title> no cabeçalho da página.

navbar: Barra de navegação da página.

content: Conteúdo principal da página.

Flask-Bootstrap

Veja como ficou o exemplo:



Flask-Bootstrap

Veja outros blocos do template-base do Flask-Bootstrap:

Bloco	Descrição	Bloco	Descrição
doc	O documento HTML completo	html_attribs	Atributos dentro da tag <html>
html	O conteúdo da tag <html>	head	O conteúdo da tag <head>
title	O conteúdo da tag <title>	metas	A lista de tags <meta>
styles	Definições de CSS	body_attribs	Atributos dentro da tag <body>
body	O conteúdo da tag <body>	navbar	Barra de navegação definida pelo usuário
content	Conteúdo de página definido pelo usuário	scripts	Declarações JavaScript no final do documento

Flask-Bootstrap

Muitos dos blocos do *template-base* são usados pelo próprio *Flask-Bootstrap*, de modo que sobrescrevê-los diretamente poderia causar problemas. Por exemplo, os blocos *styles* e *scripts* são os locais em que os arquivos *CSS* e *JavaScript* do *Bootstrap* são declarados. Se a aplicação precisar acrescentar o seu próprio conteúdo em um bloco que já tenha algum conteúdo, a função *super()* do *Jinja2* deverá ser usada.

Flask-Bootstrap

Por exemplo, é assim que o bloco scripts teria que ser escrito no template derivado para adicionar um novo arquivo JavaScript no documento:

```
{% block scripts %}
{{ super() }}
<script type="text/javascript" src="meu-script.js"></script>
{% endblock %}
```

FIM