

Mostrando capa do jogo

Transcrição

A ideia agora é pensarmos em como mostrar a imagem enviada para o servidor na tela.

Na página "Novo Jogo", o upload ainda não foi feito. Portanto, seria interessante mostrarmos um *preview* ao usuário - por exemplo, uma imagem padrão indicando que não há nenhuma imagem salva no servidor ainda.

Além disso, temos que colocar a opção de fazer upload de arquivos na página "Editar", já que os jogos que salvamos anteriormente ainda não possuem imagens associadas.

Começarmos pelo *template* `novo.html`, onde já preparamos o botão "Escolher arquivo":

```
<figure>
  <figcaption>
    <label>
      <input type="file" name="arquivo" accept=".jpg">
    </label>
  </figcaption>
</figure>
```

Dessa forma, o `<input>` servirá de legenda para nossa imagem padrão. Na estrutura `<figure></figure>`, colocaremos a imagem que queremos exibir com ``.

A imagem usada nessa aula pode ser baixada [aqui \(https://caelum-online-public.s3.amazonaws.com/739-python-flask2/03/capa_padrao.zip\)](https://caelum-online-public.s3.amazonaws.com/739-python-flask2/03/capa_padrao.zip), e deve ser colocada na pasta "uploads".

Existe uma forma bem interessante de enviar um arquivo do servidor para o *template* por meio do Flask, que é colocando-a em uma URL específica. Assim, poderemos acessá-la via `url_for()` - a melhor forma de acessar um caminho no diretório do nosso projeto.

Nesse caso, vamos definir uma rota para `imagem` passando o nome do arquivo:

```
{% extends "template.html" %}
{% block conteudo %}
  <form action="{{ url_for('criar') }}" method="post" enctype="multipart/form-data">
    <figure>
      
      <figcaption>
        <label>
          <input type="file" name="arquivo" accept=".jpg">
        </label>
      </figcaption>
    </figure>
```

Agora precisamos criar a rota que nos proverá esse acesso ao servidor. Ela precisa nos dar o caminho para `uploads` com o parâmetro `nome_arquivo`, que será passado para a função `imagem()`.

Em seguida, importaremos outra função do Flask, chamada `send_from_directory`. Ela consegue mostrar um arquivo na tela - no caso, um arquivo de um diretório específico -, e recebe como parâmetros o diretório e o nome do arquivo:

```
@app.route('/uploads/<nome_arquivo>')
def imagem(nome_arquivo):
    return send_from_directory('uploads', nome_arquivo)
```

Precisaremos fazer uma pequena alteração na página `novo.html`, já que não estamos recebendo `filename` na rota de `imagem`, mas sim `nome_arquivo`:

```
{% extends "template.html" %}
{% block conteudo %}
    <form action="{{ url_for('criar') }}" method="post" enctype="multipart/form-data">
        <figure>
            
            <figcaption>
                <label>
                    <input type="file" name="arquivo" accept=".jpg">
                </label>
            </figcaption>
        </figure>
```

Salvando essas alterações, teremos uma imagem gigantesca sendo exibida na página de "Novo Jogo":



Mas por enquanto vamos deixá-la como está. Nessa tela, faz sentido mostrar a capa padrão. Porém, na página "Editar", temos que mostrar a imagem específica do jogo que está sendo editado, e podemos fazer isso por meio do `id` do jogo. Iremos copiar o código que fizemos para mostrar a imagem em `novo.html`, mas, ao invés de `capa_padrao.jpg`, passaremos uma variável chamada `capa_jogo`.

```
{% extends "template.html" %}
{% block conteudo %}
    <form action="{{ url_for('atualizar') }}" method="post">
        <figure>
            
            <figcaption>
                <label>
                    <input type="file" name="arquivo" accept=".jpg">
                </label>
            </figcaption>
        </figure>
```

Agora vamos dar uma olhada no código que criamos para a rota `/editar` :

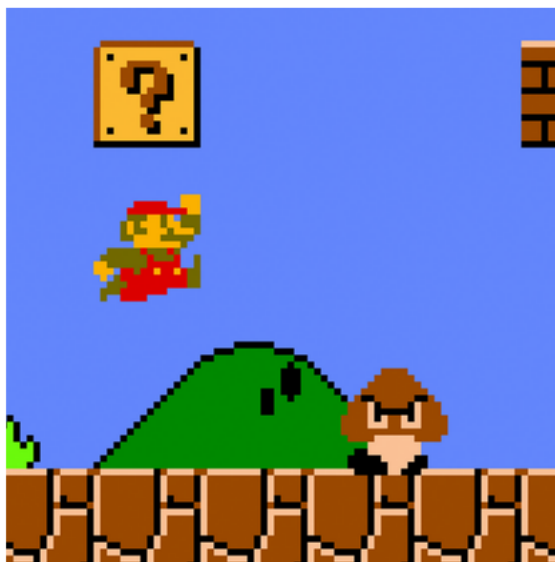
```
@app.route('/editar/<int:id>')
def editar(id):
    if 'usuario_logado' not in session or session['usuario_logado'] == None:
        return redirect(url_for('login', proxima=url_for('editar')))
    jogo = jogo_dao.busca_por_id(id)
    return render_template('editar.html', titulo='Editando jogo', jogo=jogo)
```

Nele, estamos fazendo um `render_template()` passando `editar.html` , o título "Editando jogo" e `jogo` . Além desses argumentos, passaremos também a variável `capa_jogo` recebendo `f'capa{id}.jpg'` .

```
@app.route('/editar/<int:id>')
def editar(id):
    if 'usuario_logado' not in session or session['usuario_logado'] == None:
        return redirect(url_for('login', proxima=url_for('editar')))
    jogo = jogo_dao.busca_por_id(id)
    return render_template('editar.html', titulo='Editando jogo', jogo=jogo,
                           capa_jogo = f'capa{id}.jpg')
```

Agora a tela "Editar" mostrará a imagem específica do jogo que está sendo editado:

Editando jogo

[Escolher arquivo](#) | Nenhum arquivo selecionado

Nome

Categoria

Console

Na próxima aula iremos aprimorar a usabilidade da nossa aplicação, a fim de resolver problemas como as dimensões das nossas imagens. Nos vemos lá!