

## Melhorando destino de upload

### Transcrição

Ao invés de passarmos um caminho relativo para os uploads na nossa aplicação, podemos criar, no nosso projeto, um caminho específico que será fixo para todos os arquivos de upload.

Para isso, faremos um `app.config` passando um `UPLOAD_PATH`. Como `jogoteca.py`, que é um arquivo, é a raiz da nossa aplicação, a expressão `__file__` sempre terá o valor `jogoteca.py`. Para pegarmos o caminho absoluto desse arquivo, importaremos a biblioteca `os`, que nos permite lidar com arquivos, e utilizaremos a função `path.abspath()`, passando `__file__` como parâmetro.

Em seguida, para pegarmos o nome do diretório, poderíamos fazer uma expressão regular para retirar o nome do arquivo e receber uma string. Porém, uma forma mais simples é por meio da função `path.dirname()`, que retornará justamente o nome do diretório daquele arquivo. Junto ao diretório, passaremos a pasta `/uploads`.

```
app.config['UPLOAD_PATH'] = os.path.dirname(os.path.abspath(__file__)) + '/uploads'
```

Assim, teremos o caminho padrão de todos os arquivos de imagem que queremos salvar, e poderemos acessá-lo em qualquer lugar da nossa aplicação.

Agora mudaremos o código que escrevemos em `criar()`. Primeiramente, criaremos uma variável `upload_path` que receberá `app.config['UPLOAD_PATH']`. Passaremos essa variável para o `format()` da função `save()`, concatenando o nome do arquivo (`arquivo.filename`).

```
@app.route('/criar', methods=['POST',])
def criar():
    nome = request.form['nome']
    categoria = request.form['categoria']
    console = request.form['console']
    jogo = Jogo(nome, categoria, console)
    jogo_dao.salvar(jogo)

    arquivo = request.files['arquivo']
    upload_path = app.config['UPLOAD_PATH']
    arquivo.save(f'{upload_path}/{arquivo.filename}')
    return redirect(url_for('index'))
```

Mas será que é uma boa ideia manter o mesmo nome do arquivo enviado pelo usuário? Por exemplo, se o usuário tiver dois arquivos com o nome `imagem`, mas que sejam de jogos diferentes, um irá sobrescrever o outro. Portanto, o mais indicado é criarmos um nome customizado, utilizando, por exemplo, as informações do próprio jogo.

Nesse caso, passaremos o retorno de `jogo_dao.salvar(jogo)` para uma variável `jogo`. Dessa forma, as características de `jogo` estarão acessíveis, e conseguiremos acessá-las nas funções seguintes. Em seguida criaremos o nome customizado com a palavra `capa`, `jogo.id` e a extensão `.jpg`.

```
@app.route('/criar', methods=['POST',])
def criar():
```

```
nome = request.form['nome']
categoria = request.form['categoria']
console = request.form['console']
jogo = Jogo(nome, categoria, console)
jogo = jogo_dao.salvar(jogo)

arquivo = request.files['arquivo']
upload_path = app.config['UPLOAD_PATH']
arquivo.save(f'{upload_path}/capa{jogo.id}.jpg')
return redirect(url_for('index'))
```

Dessa forma, os arquivos que enviados para o servidor por meio da nossa aplicação serão salvos corretamente na pasta especificada e com o nome customizado que implementamos. O próximo passo será criar uma forma de visualizarmos a imagem que está sendo enviada nas páginas "Novo jogo" e "Editar". Até a próxima!