Aula 04: Desenvolvimento web - Arquivos Estáticos com Express

Servindo Arquivos Estáticos com Express

Introdução

Ao desenvolver uma aplicação web, é comum precisar servir arquivos estáticos como HTML, CSS, JavaScript, imagens, e outros tipos de arquivos para que sejam acessíveis pelos navegadores dos usuários. O Express, um framework minimalista para Node.js, facilita o processo de servir esses arquivos estáticos através de seu middleware `express.static`.

O Que São Arquivos Estáticos?

Arquivos estáticos são recursos que não são gerados dinamicamente pelo servidor, ou seja, eles não mudam cada vez que são solicitados. Exemplos incluem:

- **HTML**: Estrutura da página web.
- **CSS**: Estilos aplicados à página.
- **JavaScript**: Scripts que são executados no navegador do usuário.
- **Imagens**: Arquivos como `.jpg`, `.png`, `.gif`, etc.
- **Textos**: Arquivos `.txt`, `.csv`, etc.

Esses arquivos são geralmente armazenados em uma pasta específica dentro do projeto, como `public`, `static`, ou `assets`.

Configurando o Servidor Express para Servir Arquivos Estáticos

Passo 1: Estrutura do Projeto

Primeiro, organize a estrutura do seu projeto. Uma estrutura típica pode ser assim:

```
meu-projeto/
|
├── public/
|   ├── css/
|   |   └── estilo.css
|   ├── js/
|   |   └── script.js
|   ├── imagens/
|   |   └── logo.png
|   ├── index.html
|   └── exemplo.txt
|
└── app.js
```

Passo 2: Configurando o Middleware `express.static`

Dentro do seu arquivo principal do servidor, `app.js`, configure o middleware `express.static` para servir arquivos a partir do diretório `public`:

```
const express = require('express')
const app = express()

// Middleware para servir arquivos estáticos
app.use(express.static('public'))

// Iniciando o servidor
app.listen(3000, () => {
    console.log('Servidor rodando em http://localhost:300
0')
})
```

Passo 3: Acessando os Arquivos Estáticos

Depois de configurar o middleware `express.static`, você pode acessar seus arquivos estáticos diretamente através da URL correspondente.

- **HTML**: `http://localhost:3000/index.html`
- **CSS**: `http://localhost:3000/css/estilo.css`
- **JavaScript**: `http://localhost:3000/js/script.js`
- **Imagem**: `http://localhost:3000/imagens/logo.png`
- **Texto**: `http://localhost:3000/exemplo.txt`

Servindo Arquivos Estáticos em Subdiretórios

O Express permite servir arquivos estáticos a partir de subdiretórios dentro do diretório configurado como raiz. Por exemplo, se você armazenar arquivos CSS em `public/css/`, eles estarão disponíveis em `http://localhost:3000/css/`.

Exemplo:

```
<!-- index.html -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, init
ial-scale=1.0">
    <title>Página Inicial</title>
    <link rel="stylesheet" href="/css/estilo.css">
</head>
<body>
    <h1>Bem-vindo ao meu site!</h1>
    <img src="/imagens/logo.png" alt="Logo">
    <p>Este é um exemplo de página HTML servida pelo Expres
s.</p>
    <script src="/js/script.js"></script>
</body>
</html>
```

No exemplo acima, o HTML está referenciando um arquivo CSS e uma imagem que estão localizados em subdiretórios do diretório `public`.

Definindo Múltiplos Diretórios de Arquivos Estáticos

Você pode definir mais de um diretório para servir arquivos estáticos, se necessário. Para isso, basta chamar `express.static` várias vezes com caminhos diferentes:

```
app.use(express.static('public'))
app.use(express.static('assets'))
```

Agora, os arquivos estáticos podem ser servidos a partir de ambos os diretórios `public` e `assets`.

Personalizando o Caminho Virtual

Você também pode definir um "caminho virtual" que será usado na URL para acessar os arquivos estáticos. Por exemplo, se você quiser que todos os arquivos estáticos estejam disponíveis em um subcaminho como `/static`, você pode configurar o `express.static` assim:

```
app.use('/static', express.static('public'))
```

Agora, para acessar `public/index.html`, você usaria a URL `http://localhost:3000/static/index.html`.

Cache de Arquivos Estáticos

Por padrão, o Express habilita o cache para arquivos estáticos, o que ajuda a melhorar a performance da sua aplicação. Se necessário, você pode configurar as opções de cache ao usar o `express.static`:

```
app.use(express.static('public', {
    maxAge: '1d'  // Cache de 1 dia
}))
```

Conclusão

Servir arquivos estáticos é uma parte fundamental do desenvolvimento web, e o Express torna essa tarefa simples e eficiente com seu middleware `express.static`. Com essa abordagem, você pode facilmente disponibilizar seus recursos como HTML, CSS, JavaScript, imagens, e outros arquivos diretamente para os usuários.

Este conhecimento é crucial para criar aplicações web robustas, onde os recursos estáticos precisam ser acessados de maneira rápida e segura.