Aula 06: sobre EJS (Embedded JavaScript)

---

1. Introdução ao EJS

- **O que é EJS?**
  - EJS (Embedded JavaScript) é uma **template engine** usada no Node.js para gerar HTML dinâmico com JavaScript embutido.
  - Permite a criação de páginas HTML dinâmicas com código JavaScript inserido diretamente no HTML.
  - Uma das principais vantagens é a facilidade de uso, semelhante a outras linguagens de template como **JSP** ou **PHP**.
- **Quando usar o EJS?**
  - Quando você precisa de conteúdo dinâmico em suas páginas HTML, como exibir dados de um banco de dados ou manipular formulários.
  - É útil em aplicações web tradicionais, onde o servidor renderiza as páginas no lado do servidor.

---

2. Configurando o Projeto Node.js com EJS

1. **Instalando o EJS**
   - Crie um novo projeto Node.js:

```
mkdir projeto-ejs
cd projeto-ejs
npm init -y
```

   - Instale o Express e o EJS:

```
npm install express ejs
```

2. **Configurando o EJS no Express**
   - Para utilizar o EJS como motor de templates no Express, é necessário definir algumas configurações:

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

// Configurar o EJS como template engine
app.set('view engine', 'ejs');

// Definir o diretório de views (opcional, por padrão é /views)
app.set('views', './views');

app.listen(3000, () => {
    console.log('Servidor rodando na porta 3000');
});
```

---

3. Estrutura de Diretórios e Arquivos

- **Pasta `/views`**:
  - A pasta onde os arquivos EJS ficarão armazenados. Por padrão, o Express busca os templates na pasta `views`.

Exemplo de Estrutura:

```
projeto-ejs/
├── views/
│   ├── index.ejs
│   ├── about.ejs
└── app.js
```

- **Arquivo de exemplo (`index.ejs`):**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, i
nitial-scale=1.0">
    <title>Home Page</title>
</head>
<body>
    <h1>Bem-vindo, <%= nome %>!</h1>
    <p>Data atual: <%= data %></p>
</body>
</html>
```

---

4. Sintaxe Básica do EJS

1. **Inserindo Variáveis**
   - Utiliza-se `<%= %>` para imprimir variáveis no HTML.

```
<p>Olá, <%= nome %>!</p>
```

2. **Lógica de Controle**
   - Condicionais (`if`, `else`):

```
<% if (idade >= 18) { %>
    <p>Você é maior de idade.</p>
<% } else { %>
    <p>Você é menor de idade.</p>
```

```
<% } %>
```

- Loops (`for`, `forEach`):

```
<ul>
  <% lista.forEach(item => { %>
      <li><%= item %></li>
  <% }) %>
</ul>
```

3. **Comentários**
   - Comentários no EJS não são exibidos no HTML renderizado.

```
<%-- Este é um comentário no EJS --%>
```

---

5. Renderizando Páginas com Dados Dinâmicos

1. **Renderizando uma página EJS no Express:**
   - No arquivo `app.js`, crie uma rota que renderiza o arquivo EJS:

```
app.get('/', (req, res) => {
    const dados = {
        nome: 'Marcos',
        data: new Date().toLocaleDateString()
    };
    res.render('index', dados);
});
```

2. **Passando dados para o template:**

- No exemplo acima, estamos passando o objeto `dados` para o template `index.ejs`, e as variáveis `nome` e `data` serão usadas no HTML dinâmico.

---

6. Layouts e Reuso de Código com EJS

1. **Includando Arquivos**
   - Uma das funcionalidades mais importantes do EJS é a capacidade de incluir partes do HTML, como cabeçalhos, rodapés, etc.

```
<%- include('header') %>
<h1>Conteúdo Principal</h1>
<%- include('footer') %>
```

   - Arquivo `header.ejs`:

```
<header>
    <h1>Cabeçalho do Site</h1>
</header>
```

   - Arquivo `footer.ejs`:

```
<footer>
    <p>Rodapé do Site</p>
</footer>
```

2. **Exemplo Completo de Reuso:**
   - Suponha que você queira usar o mesmo cabeçalho e rodapé em várias páginas. Inclua-os em todas as páginas que precisar, mantendo o código mais organizado e modular.

---

7. Manipulando Formulários com EJS

1. **Criando um Formulário HTML**
   - Suponha que você tenha uma página com um formulário de contato:

```
<form action="/contato" method="POST">
    <input type="text" name="nome" placeholder="Nome">
    <input type="email" name="email" placeholder="Emai
l">
    <button type="submit">Enviar</button>
</form>
```

2. **Capturando Dados no Servidor**
   - No Express, você pode capturar os dados do formulário assim:

```
app.post('/contato', (req, res) => {
    const { nome, email } = req.body;
    res.send(`Formulário recebido! Nome: ${nome}, Email:
${email}`);
});
```

   - **Lembre-se** de usar middlewares como `express.urlencoded()` para capturar os dados enviados pelo formulário:

```
app.use(express.urlencoded({ extended: true }));
```

---

8. Trabalhando com Partials e Layouts

- **Partials**: São pequenos pedaços de código EJS reutilizáveis. Você pode incluir partials dentro de outras páginas EJS para evitar duplicação de código.

- Por exemplo, o cabeçalho e o rodapé podem ser partials que são incluídos em várias páginas.

- **Exemplo de Partial (`header.ejs`)**:

```
<header>
    <h1>Meu Site</h1>
</header>
```

- **Incluindo o Partial na Página Principal**:

```
<%- include('header') %>
<h2>Página Inicial</h2>
<%- include('footer') %>
```

---

9. Boas Práticas e Considerações Finais

1. **Modularize seu Código**
   - Utilize partials para evitar duplicação de código.
   - Mantenha seu código EJS limpo, evitando lógica de negócio dentro dos templates. A lógica deve estar no servidor, e o EJS deve ser usado apenas para exibir os dados.
2. **Segurança**
   - EJS automaticamente escapa variáveis para evitar **ataques XSS** (Cross-Site Scripting). Sempre utilize `<%= %>` ao exibir conteúdo dinâmico que vem de fontes externas.
3. **Performance**
   - O uso do EJS é eficiente, mas para aplicações com grande escala ou alto desempenho, considere técnicas como **caching** ou migrar para soluções baseadas em APIs e frameworks frontend, como React ou Vue.js, onde o lado do cliente renderiza o conteúdo dinâmico.

Conclusão

- O EJS é uma ferramenta simples, poderosa e fácil de aprender para criar interfaces dinâmicas no Node.js.
- Ele é particularmente útil em projetos que utilizam o render no lado do servidor para gerar HTML com base em dados dinâmicos.
- Ao utilizar EJS, é importante modularizar seu código e aplicar boas práticas de organização e segurança.

Exercícios Práticos

1. **Exercício 1**: Crie uma página EJS que exiba a data atual e uma saudação personalizada.
2. **Exercício 2**: Crie uma aplicação Express com EJS que receba dados de um formulário e exiba esses dados em uma nova página.
3. **Exercício 3**: Usando partials, crie uma página com um cabeçalho e rodapé reutilizáveis.