Guía Completa: Uso de EJS con `express-ejs-layouts` para Layouts, Parte 1

Introducción

¿Qué es EJS?

EJS (*Embedded JavaScript*) es un motor de plantillas que permite combinar HTML con JavaScript para generar contenido dinámico en el servidor. Es ideal para aplicaciones Node.js que necesitan personalizar sus vistas.

¿Qué es `express-ejs-layouts`?

`express-ejs-layouts` es un middleware que facilita la implementación de layouts en EJS. Con esta herramienta, puedes definir una estructura base que se reutiliza en todas las vistas, manteniendo el código limpio y organizado.

Ventajas de EJS con `express-ejs-layouts`

1. **Facilidad de configuración**: Simplifica el uso de layouts.
2. **Reutilización de código**: Centraliza la estructura principal en un solo archivo.
3. **Compatibilidad con partials**: Sigue permitiendo el uso de componentes reutilizables como navbar y footer.

Estructura del Proyecto

📂 **ejs-layout-example** *(Carpeta principal del proyecto)*
├── 📁 **views/** *(Carpeta para las vistas EJS)*
│ ├── 🧩 **partials/** *(Componentes reutilizables)*
│ │ ├── 🟦 **navbar.ejs** *(Barra de navegación con iconos y enlaces)*
│ │ ├── 🟩 **footer.ejs** *(Pie de página con derechos de autor)*
│ ├── 📝 **index.ejs** *(Vista principal)*
│ ├── 🎨 **layout.ejs** *(Estructura base del layout)*
├── 📁 **public/** *(Carpeta para archivos estáticos)*
│ ├── 🖼️ **images/** *(Imágenes como el logo)*
│ │ ├── 🌟 **logo.png** *(Logo del sitio web)*

│ ├── 🎨 **styles.css** *(Hoja de estilos personalizada)*
│ ├── 📁 **icons/** *(Iconos para el navbar)*
│ │ ├── 🏠 **home.svg** *(Icono de inicio)*
│ │ ├── ℹ️ **info.svg** *(Icono de información)*
│ │ ├── ✉️ **contact.svg** *(Icono de contacto)*
├── ⚙️ **app.mjs** *(Archivo principal del servidor)*
├── 📦 **package.json** *(Configuración del proyecto Node.js)*

Paso 1: Configurar el Proyecto

1. Crea la carpeta del proyecto y configura Node.js:

```
mkdir ejs-layout-example
cd ejs-layout-example
npm init -y
```

2. Instala las dependencias necesarias:

```
npm install express ejs express-ejs-layouts
```

3. Crea las carpetas y archivos principales:

```
mkdir views public public/images public/icons
touch app.mjs public/styles.css
```

Paso 2: Configurar el Servidor con `express-ejs-layouts`

Archivo Principal del Servidor (`app.mjs`)

```
import express from 'express';
import path from 'path';
import expressLayouts from 'express-ejs-layouts';

const app = express();
const PORT = 3000;

// Configurar EJS como motor de plantillas
app.set('view engine', 'ejs');
app.set('views', path.resolve('./views'));
```

```
// Configurar express-ejs-layouts
app.use(expressLayouts);
app.set('layout', 'layout'); // Archivo base de layout

// Servir archivos estáticos
app.use(express.static(path.resolve('./public')));

// Ruta principal
app.get('/', (req, res) => {
    res.render('index', {
        title: 'Página Principal',
        navbarLinks: [
            { text: 'Inicio', href: '/', icon: '/icons/home.svg' },
            { text: 'Acerca de', href: '/about', icon: '/icons/info.svg' },
            { text: 'Contacto', href: '/contact', icon: '/icons/contact.svg' }
        ]
    });
});

// Iniciar el servidor
app.listen(PORT, () => {
    console.log(`Servidor ejecutándose en http://localhost:${PORT}`);
});
```

Explicación:

1. **Importar** `express-ejs-layouts`:

   ```
   import expressLayouts from 'express-ejs-layouts';
   ```

2. **Activar el middleware**:

   ```
   app.use(expressLayouts);
   ```

3. **Configurar el archivo de layout**:

   ```
   app.set('layout', 'layout');
   ```

   Esto indica que el archivo base para los layouts será `layout.ejs`.

Paso 3: Crear las Vistas y Componentes

Layout Base (`views/layout.ejs`)

Define la estructura principal del sitio web:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title><%= title %></title>
    <link rel="stylesheet" href="/styles.css">
</head>
<body>
    <!-- Navbar -->
    <%- include('partials/navbar') %>

    <!-- Contenido dinámico -->
    <main>
        <%- body %>
    </main>

    <!-- Footer -->
    <%- include('partials/footer') %>
</body>
</html>
```

Explicación:

- `<%- include('partials/navbar') %>`: Incluye el componente del navbar.
- `<%- body %>`: Espacio reservado para el contenido dinámico de cada vista.
- `<%- include('partials/footer') %>`: Incluye el footer.

Navbar con Iconos y Logo (`views/partials/navbar.ejs`)

```
<nav>
    <div class="navbar">
        <a href="/">
            <img src="/images/logo.png" alt="Logo" class="logo">
        </a>
        <ul>
            <% navbarLinks.forEach(link => { %>
                <li>
                    <a href="<%= link.href %>">
                        <img src="<%= link.icon %>" alt="<%= link.text %> Icon">
                        <%= link.text %>
                    </a>
                </li>
```

```
            <% }); %>
        </ul>
    </div>
</nav>
```

Explicación:

- **Logo**:

  ```
  <img src="/images/logo.png">
  ```

  Muestra un logo dinámico.

- **Enlaces dinámicos con iconos**:

  ```
  <% navbarLinks.forEach(link => { %>
  ```

  Genera los enlaces dinámicamente a partir de los datos enviados desde el servidor.

Footer (`views/partials/footer.ejs`)

```
<footer>
    <p>&copy; Modulo 3 - EJS Layout P1 - by Nodo Tecnologico - 2024</p>
</footer>
```

Explicación:

- **Contenido fijo**: Contiene un mensaje con derechos de autor.

Página Principal (`views/index.ejs`)

```
<section>
    <h1>Bienvenido a mi sitio web</h1>
    <p>Este es un ejemplo de cómo usar layouts en EJS con `express-ejs-layouts`.</p>
</section>
```

Explicación:

- **Contenido dinámico**: Este archivo define el contenido que se mostrará en el espacio reservado por `<%- body %>`.

Paso 4: Agregar Estilos

Archivo de Estilos (`public/styles.css`)

```css
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
}

.navbar {
    display: flex;
    align-items: center;
    background-color: #333;
    padding: 10px 20px;
    color: white;
}

.navbar .logo {
    width: 50px;
    margin-right: 20px;
}

.navbar ul {
    list-style: none;
    display: flex;
    gap: 20px;
    margin: 0;
    padding: 0;
}

.navbar ul li a {
    color: white;
    text-decoration: none;
    display: flex;
    align-items: center;
    gap: 5px;
}

.navbar ul li a img {
    width: 20px;
}

footer {
    text-align: center;
    padding: 10px;
    background-color: #333;
    color: white;
    position: fixed;
```

```
    bottom: 0;
    width: 100%;
}
```

Explicación:

- **Navbar**: Usa `flexbox` para alinear el logo y los enlaces.
- **Footer**: Fijo al final de la página con un diseño oscuro.

Paso 5: Ejecutar el Proyecto

1. Inicia el servidor:

   ```
   node --watch app.mjs
   ```

2. Abre en tu navegador:

   ```
   http://localhost:3000
   ```

Resultado Final

- **Navbar**: Incluye un logo, enlaces dinámicos e iconos.
- **Footer**: Contiene derechos de autor.
- **Layout Base**: Usa `express-ejs-layouts` para simplificar la estructura del proyecto.

Guía Completa: Uso de EJS con `express-ejs-layouts` para Layouts, Parte 2

Actualización: Datos del Navbar Hardcodeados y Nuevas Páginas

Se eliminará el paso de pasar los datos del navbar como valores dinámicos en el `app.mjs`, ya que siempre tendrán los mismos elementos. Además, se agregarán las vistas para las páginas de “Acerca de” e “Inicio”.

Estructura del Proyecto Actualizada

📁 **ejs-layout-example** *(Carpeta principal del proyecto)*
├── 📁 **views/** *(Carpeta para las vistas EJS)*
│ ├── 🧩 **partials/** *(Componentes reutilizables)*
│ │ ├── 🟦 **navbar.ejs** *(Barra de navegación con iconos y enlaces)*
│ │ ├── 🟩 **footer.ejs** *(Pie de página con derechos de autor)*
│ ├── 📝 **index.ejs** *(Vista principal)*
│ ├── ℹ️ **about.ejs** *(Página de Acerca de)*
│ ├── ✉️ **contact.ejs** *(Página de Contacto)*
│ ├── 🎨 **layout.ejs** *(Estructura base del layout)*
├── 📁 **public/** *(Carpeta para archivos estáticos)*
│ ├── 🖼️ **images/** *(Imágenes como el logo)*
│ │ ├── 🌟 **logo.png** *(Logo del sitio web)*
│ ├── 🎨 **styles.css** *(Hoja de estilos personalizada)*
│ ├── 📁 **icons/** *(Iconos para el navbar)*
│ │ ├── 🏠 **home.svg** *(Icono de inicio)*
│ │ ├── ℹ️ **info.svg** *(Icono de información)*
│ │ ├── ✉️ **contact.svg** *(Icono de contacto)*
├── ⚙️ **app.mjs** *(Archivo principal del servidor)*
├── 📦 **package.json** *(Configuración del proyecto Node.js)*

Paso 1: Actualizar el Servidor (`app.mjs`)

Elimina la lógica para pasar los datos del navbar. Ahora, cada página será configurada directamente en sus respectivas rutas:

```
import express from 'express';
import path from 'path';
import expressLayouts from 'express-ejs-layouts';

const app = express();
const PORT = 3000;

// Configurar EJS como motor de plantillas
app.set('view engine', 'ejs');
app.set('views', path.resolve('./views'));

// Configurar express-ejs-layouts
app.use(expressLayouts);
app.set('layout', 'layout'); // Archivo base de layout

// Servir archivos estáticos
app.use(express.static(path.resolve('./public')));
```

```
// Ruta principal
app.get('/', (req, res) => {
    res.render('index', { title: 'Página Principal' });
});

// Ruta para la página Acerca de
app.get('/about', (req, res) => {
    res.render('about', { title: 'Acerca de Nosotros' });
});

// Ruta para la página de Contacto
app.get('/contact', (req, res) => {
    res.render('contact', { title: 'Contáctanos' });
});

// Iniciar el servidor
app.listen(PORT, () => {
    console.log(`Servidor ejecutándose en http://localhost:${PORT}`);
});
```

Explicación de cambios:

1. **Datos hardcodeados en el navbar**: Los enlaces no se pasan como valores dinámicos, sino que están definidos directamente en el archivo `navbar.ejs`.
2. **Rutas nuevas**:
    - `/about`: Renderiza la página “Acerca de”.
    - `/contact`: Renderiza la página “Contacto”.

Paso 2: Navbar Hardcodeado (`views/partials/navbar.ejs`)

Elimina el uso de valores dinámicos y define los enlaces directamente en el archivo:

```
<nav>
    <div class="navbar">
        <a href="/">
            <img src="/images/logo.png" alt="Logo" class="logo">
        </a>
        <ul>
            <li>
                <a href="/">
                    <img src="/icons/home.svg" alt="Inicio Icon">
                    Inicio
                </a>
            </li>
```

```
            <li>
                <a href="/about">
                    <img src="/icons/info.svg" alt="Acerca de Icon">
                    Acerca de
                </a>
            </li>
            <li>
                <a href="/contact">
                    <img src="/icons/contact.svg" alt="Contacto Icon">
                    Contacto
                </a>
            </li>
        </ul>
    </div>
</nav>
```

Explicación:

1. **Estructura fija del navbar**: Los enlaces ya no dependen de datos dinámicos. Esto simplifica la lógica del servidor.
2. **Iconos y enlaces**: Cada enlace incluye un icono SVG para mejorar la experiencia visual.

Paso 3: Crear las Nuevas Vistas

Página "Acerca de" (`views/about.ejs`)

```
<section>
    <h1>Acerca de Nosotros</h1>
    <p>Somos un equipo Docente empeñados en sacar a relucir el potencial de cada alumno,
        a base de practicas guiadas</p>
</section>
```

Explicación:

- Usa el layout base y define el contenido dinámico para la sección principal.

Página "Contacto" (`views/contact.ejs`)

```
<section>
    <h1>Contáctanos</h1>
    <p>Puedes comunicarte con nosotros a través del correo
```

```
        <a  href="mailto:ingmartinmsalas@gmail.com">ingmartinmsalas@gmail.com</a>.</p>
</section>
```

Explicación:

- El contenido se centra en un mensaje y un enlace de correo electrónico.

Paso 4: Página Principal (`views/index.ejs`)

```
<section>
    <h1>Bienvenido a nuestro sitio web</h1>
    <p>Este es un ejemplo de cómo usar layouts en EJS con `express-ejs-layouts`.</p>
</section>
```

Explicación:

- La página principal sigue el mismo formato, pero con contenido diferente.

Paso 5: Verificar la Aplicación

1. Inicia el servidor:

   ```
   node --watch app.mjs
   ```

2. Visita las siguientes URLs en tu navegador:

   - **Inicio**: http://localhost:3000/
   - **Acerca de**: http://localhost:3000/about
   - **Contacto**: http://localhost:3000/contact

Resultado Final

1. **Navbar**:
   - Contiene enlaces a “Inicio”, “Acerca de” y “Contacto”, con iconos y un logo.
2. **Footer**:

    - Muestra los derechos de autor en todas las páginas.
3. **Páginas Dinámicas**:
    - Cada ruta muestra contenido personalizado.

¡Felicidades! Ahora tienes un sitio web completamente funcional con EJS, layouts y varias páginas dinámicas.