Guía: Configuración de ESLint con la Guía de Estilo de Airbnb

Paso 1: Inicializar un Proyecto de Node.js

1. Abre la terminal en la carpeta donde deseas crear tu proyecto.
2. Ejecuta el siguiente comando para inicializar un proyecto de Node.js y crear un archivo `package.json`:

   npm init -y

Paso 2: Instalar ESLint con la Configuración de Airbnb

1. Instala ESLint junto con la configuración de Airbnb y sus dependencias necesarias:

   npx install-peerdeps --dev eslint-config-airbnb

Paso 3: Configurar ESLint

1. Inicializa la configuración de ESLint ejecutando el siguiente comando y sigue las instrucciones:

   npx eslint --init

Paso 4: Ajustar el Archivo `eslint.config.mjs`

1. **Crea o abre el archivo** `eslint.config.mjs` y asegúrate de que contenga la siguiente configuración:

```
import globals from "globals";
import pluginJs from "@eslint/js";

export default [
  {
    languageOptions: {
      ecmaVersion: "latest", // Asegura que se utilice la última versión de ECMAScript
      sourceType: "module", // Soporte para ES Modules
      globals: globals.node, // Globales específicos de Node.js
    },
    plugins: {
      js: pluginJs,
    },
    rules: {
      "semi": ["error", "always"], // Requiere punto y coma al final de las líneas
      "quotes": ["error", "double"], // Requiere comillas dobles para strings
      "no-var": "error", // Prohibir el uso de `var`
      "prefer-const": "error", // Prefiere `const` si la variable no es reasignada
      "indent": ["error", 2], // Indentación de 2 espacios
    },
  },
];
```

Paso 5: Configurar Visual Studio Code para Correcciones Automáticas

1. **Abre Visual Studio Code**.
2. **Accede al archivo** `settings.json`:
   - Presiona `Ctrl + ,` para abrir la configuración.
   - Haz clic en el ícono de engranaje en la esquina superior derecha para abrir la configuración en formato JSON.
3. **Añade la siguiente configuración** para habilitar las correcciones automáticas de ESLint al guardar:

```
{
  "workbench.iconTheme": "material-icon-theme",
  "security.workspace.trust.untrustedFiles": "open",
  "workbench.editorAssociations": {
    "*.docx": "default"
  },
  "eslint.run": "onSave",
  "eslint.codeActionsOnSave.rules": null,
  "editor.codeActionsOnSave": {
    "source.fixAll.eslint": "explicit"
  },
  "eslint.validate": [
    "javascript",
    "javascriptreact",
    "typescript",
    "typescriptreact"
  ],
  "eslint.execArgv": null
}
```

Paso 6: Ejecutar y Probar ESLint

1. **Introduce algunos errores de estilo** en un archivo JavaScript (por ejemplo, `index.js`).
2. **Guarda el archivo** para ver si ESLint aplica las correcciones automáticamente.
3. **Ejecuta ESLint manualmente** para corregir los errores:

```
npx eslint index.js –fix
```

Subir archivos a Github

1. **Inicializa el repositorio localmente (si aún no lo has hecho)**
   git init
2. **Añade todos los archivos a la zona de preparación (staging area)**
   git add .
3. **Haz un commit de los archivos**
   git commit -m "Mensaje del commit"

4. Vincula tu repositorio local con el repositorio remoto de GitHub

Este comando solo lo necesitas si aún no has vinculado tu repositorio local al remoto:

git remote add origin https://github.com/tu-usuario/tu-repo.git

5. Sube los cambios al repositorio remoto

⑩ **Si quieres subir los cambios a la rama** `main` **(asumiendo que esa es la rama principal):**

```
git push -u origin main
```

⑩ **Si tu rama principal se llama** `master`**:**

```
git push -u origin master
```

Nota: Si ya has subido previamente commits a este repositorio y solo estás agregando nuevos cambios, puedes utilizar simplemente:

```
git push
```

Opcional) Paso 7: Integrar Husky para Prevenir Errores en Commits

1. **Instala Husky:**

```
npm install husky --save-dev
```

2. **Configura Husky:**

```
npx husky install
```

3. **Añade un hook para correr ESLint antes de cada commit:**

```
npx husky add .husky/pre-commit "npx eslint ."
```

4. **Opcional: Configura** `lint-staged` **para que ESLint solo se ejecute en los archivos que se han modificado:**

```
npm install lint-staged --save-dev
```

Agrega en package. json:

```
json
Copiar código
"lint-staged": {
  "*.js": "eslint --fix"
}
```

Modifica el hook de Husky:

```
npx husky add .husky/pre-commit "npx lint-staged"
```

Verifica el estado del repositorio:

- ❿ Ejecuta el siguiente comando en la terminal para ver qué archivos han sido modificados o agregados:

```
git status
```

- ❿ **Añade todos los archivos modificados y agregados a la zona de preparación:**
- ❿ Para añadir todos los archivos que han cambiado, utiliza:

```
git add .
```

- ❿ Esto añadirá todos los archivos modificados, nuevos o eliminados al área de preparación.
- ❿ **Realiza un commit de los cambios:**
- ❿ Crea un commit con un mensaje que describa los cambios realizados:

```
git commit -m "Descripción de los cambios realizados"
```

- ❿ **Empuja los cambios al repositorio remoto:**
- ❿ Para enviar los cambios a la rama principal main en GitHub, usa:

```
git push origin master
```