最终推荐配置

为了实现高效的可视化编辑和实时预览，建议使用以下组合：

1. VS Code 编辑器:

安装 VS Code 并配置相关插件。

2. 插件列表:

JSX Preview: 实时预览 JSX 代码（轻量级且集成度高）。

ESLint: 代码检查工具。

Prettier: 代码格式化工具。

Live Server: 启动本地开发服务器，实现实时预览。

安装步骤

1. 安装 VS Code:

下载并安装 [Visual Studio Code](https://code.visualstudio.com/)。

2. 安装插件:

打开 VS Code。

进入扩展市场（可以通过左侧活动栏中的 Extensions 图标或按 `Ctrl+Shift+X`）。

搜索并安装以下插件：

JSX Preview

ESLint

Prettier Code formatter

Live Server

3. 配置项目:

创建一个新的 React 项目（如果还没有的话），可以使用 Create React App：

```bash

npx createreactapp myapp

cd myapp

```

将你的 `.jsx` 文件放入 `src` 目录中。

4. 启动 Live Server:

在 VS Code 中打开项目文件夹。

右键点击 `public/index.html` 文件，选择 `Open with Live Server`。

5. 使用 JSX Preview:

打开你的 `.jsx` 文件。

使用 JSX Preview 插件查看实时预览。

6. 配置 ESLint 和 Prettier:

安装 ESLint 和 Prettier 依赖：

```bash

npm install eslint prettier savedev

```

添加 ESLint 配置文件（`.eslintrc.js`）：

```javascript

module.exports = {

extends: [

'eslint:recommended',

'plugin:react/recommended',

'prettier'

],

plugins: ['react'],

parserOptions: {

ecmaVersion: 2020,

sourceType: 'module',

ecmaFeatures: {

jsx: true

}

},

settings: {

react: {

version: 'detect'

}

},

rules: {

// 自定义规则

}

};

```

添加 Prettier 配置文件（`.prettierrc`）：

```json

{

"singleQuote": true,

"semi": false,

"trailingComma": "all",

"printWidth": 80

}

```

7. 集成 ESLint 和 Prettier:

安装 `eslintconfigprettier` 和 `eslintpluginprettier`：

```bash

npm install eslintconfigprettier eslintpluginprettier savedev

```

更新 `.eslintrc.js` 以集成 Prettier：

```javascript

module.exports = {

extends: [

'eslint:recommended',

'plugin:react/recommended',

'plugin:prettier/recommended'

],

plugins: ['react', 'prettier'],

parserOptions: {

ecmaVersion: 2020,

sourceType: 'module',

ecmaFeatures: {

jsx: true

}

},

settings: {

react: {

version: 'detect'

}

},

rules: {

'prettier/prettier': 'error'

}

};

```

完整配置总结

项目结构

```

myapp/

├── node\_modules/

├── public/

│ ├── assets/

│ │ └── images/

│ │ ├── background1.jpg

│ │ ├── background2.jpg

│ │ └── background3.jpg

│ ├── index.html

│ └── ...

├── src/

│ ├── App.css

│ ├── App.jsx

│ ├── index.css

│ └── index.jsx

├── .eslintrc.js

├── .prettierrc

├── package.json

└── ...

```

`App.jsx`

```jsx

import React, { useState } from 'react';

import './App.css'; // 引入合并后的CSS文件

const App = () => {

const [hovered, setHovered] = useState(false);

const [overlay, setOverlay] = useState(null);

const handleMouseEnter = () => {

setHovered(true);

};

const handleMouseLeave = () => {

setHovered(false);

};

const handleOptionClick = (option) => {

setOverlay(option);

};

const closeOverlay = () => {

setOverlay(null);

};

return (

<div className="appcontainer">

<div

className="buttoncontainer"

onMouseEnter={handleMouseEnter}

onMouseLeave={handleMouseLeave}

>

<button className="mainbutton">Hover me</button>

{hovered && (

<div className="optionscontainer">

<button onClick={() => handleOptionClick('option1')}>Option 1</button>

<button onClick={() => handleOptionClick('option2')}>Option 2</button>

<button onClick={() => handleOptionClick('option3')}>Option 3</button>

</div>

)}

</div>

{overlay === 'option1' && (

<div className="overlay">

<div className="overlaycontent">

<img src="/assets/images/background1.jpg" alt="Background" />

<h1>Option 1 Content</h1>

<p>This is the content for option 1.</p>

<button onClick={closeOverlay}>Close</button>

</div>

</div>

)}

{overlay === 'option2' && (

<div className="overlay">

<div className="overlaycontent">

<img src="/assets/images/background2.jpg" alt="Background" />

<h1>Option 2 Content</h1>

<p>This is the content for option 2.</p>

<button onClick={closeOverlay}>Close</button>

</div>

</div>

)}

{overlay === 'option3' && (

<div className="overlay">

<div className="overlaycontent">

<img src="/assets/images/background3.jpg" alt="Background" />

<h1>Option 3 Content</h1>

<p>This is the content for option 3.</p>

<button onClick={closeOverlay}>Close</button>

</div>

</div>

)}

</div>

);

};

export default App;

```

`App.css`

```css

/ 原来的 app.css 内容 /

body {

fontfamily: Arial, sansserif;

margin: 0;

padding: 0;

}

.container {

display: flex;

justifycontent: center;

alignitems: center;

height: 100vh;

}

/ 新增的样式 /

.appcontainer {

position: relative;

height: 100vh;

}

.buttoncontainer {

position: absolute;

top: 0;

left: 0;

}

.mainbutton {

padding: 10px 20px;

fontsize: 16px;

cursor: pointer;

}

.optionscontainer {

position: absolute;

top: calc(100% + 5px); / 调整子按钮的位置 /

left: 0;

backgroundcolor: white;

border: 1px solid ccc;

boxshadow: 0 2px 5px rgba(0, 0, 0, 0.2);

zindex: 1;

}

.optionscontainer button {

width: 100%;

padding: 10px;

textalign: left;

border: none;

background: none;

cursor: pointer;

}

.optionscontainer button:hover {

backgroundcolor: f0f0f0;

}

.overlay {

position: fixed;

top: 0;

left: 0;

right: 0;

bottom: 0;

backgroundcolor: rgba(0, 0, 0, 0.7);

display: flex;

justifycontent: center;

alignitems: center;

zindex: 2;

}

.overlaycontent {

position: relative;

backgroundcolor: white;

padding: 20px;

borderradius: 8px;

maxwidth: 80%;

maxheight: 80%;

overflowy: auto;

boxshadow: 0 2px 10px rgba(0, 0, 0, 0.3);

}

.overlaycontent img {

width: 100%;

height: auto;

bordertopleftradius: 8px;

bordertoprightradius: 8px;

}

.overlaycontent h1 {

margintop: 0;

}

.overlaycontent button {

position: absolute;

top: 10px;

right: 10px;

padding: 5px 10px;

backgroundcolor: red;

color: white;

border: none;

borderradius: 4px;

cursor: pointer;

}

.overlaycontent button:hover {

backgroundcolor: darkred;

}

```

`.eslintrc.js`

```javascript

module.exports = {

extends: [

'eslint:recommended',

'plugin:react/recommended',

'plugin:prettier/recommended'

],

plugins: ['react', 'prettier'],

parserOptions: {

ecmaVersion: 2020,

sourceType: 'module',

ecmaFeatures: {

jsx: true

}

},

settings: {

react: {

version: 'detect'

}

},

rules: {

'prettier/prettier': 'error'

}

};

```

`.prettierrc`

```json

{

"singleQuote": true,

"semi": false,

"trailingComma": "all",

"printWidth": 80

}

```

总结

JSX Preview: 提供实时预览，适合在编辑器内部查看效果。

ESLint: 代码检查工具，确保代码质量。

Prettier: 代码格式化工具，保持代码风格一致。

Live Server: 启动本地开发服务器，实现实时预览效果。

通过这种配置，你可以在 Windows 系统上高效地进行 `.jsx` 文件的可视化编辑和实时预览。