CAFÉ, DESENVOLVIMENTO & VUE.JS

VUE.JS

Boas maneiras de organizar o projeto VUE.js

Paulo Henrique Amigoni

www.amigoni.com.br
paulo.amigoni@gmail.com

linkedin.com/in/pauloamigoni/
github.com/pauloamigoni/

CONHECENDO

VUE.JS

Boas maneiras de organizar o projeto Vue.js

Pode ser difícil manter um grande projeto Vue.js estruturado, mas há várias táticas que você pode empregar para fazê-lo. Aqui estão algumas dicas úteis para estruturar um projeto considerável do Vue.js:

Use uma arquitetura modular

Uma arquitetura modular é uma abordagem de design de software que enfatiza a quebra de um sistema grande em módulos ou componentes menores e independentes que podem ser facilmente mantidos, testados e reutilizados. No contexto de um projeto Vue.js, uma arquitetura modular envolve a divisão do seu aplicativo em componentes menores e reutilizáveis que podem ser facilmente gerenciados e dimensionados.
Aqui estão algumas práticas recomendadas a serem seguidas ao usar uma arquitetura modular no seu projeto Vue.js:

1. Use o Vuex para gerenciar o estado: O Vuex é uma biblioteca de gerenciamento de estado que permite centralizar e gerenciar o estado do seu aplicativo de maneira previsível. Seu aplicativo pode ser dividido em módulos menores e reutilizáveis com o Vuex, simples de testar e gerenciar.

2. Utilize o Nuxt.js para renderização no lado do servidor: os aplicativos Vue.js podem ser criados usando a estrutura de renderização do lado do servidor do Nuxt.js. Você pode segmentar seu aplicativo em módulos menores e reutilizáveis, simples de testar e manter, utilizando o Nuxt.js.

3. Utilize componentes para elementos de interface do usuário: Empregue componentes para separar sua interface do usuário em elementos mais gerenciáveis, escaláveis e menores. Ao fazer isso, você pode impedir a duplicação de código e garantir a consistência em todo o seu programa.

4. Usar plugins para fornecer funcionalidade reutilizável ao seu programa, como bibliotecas de terceiros ou utilitários exclusivos, é uma boa ideia. Você pode manter um código de aplicativo bem organizado e limpo usando plug-ins.

5. Para funcionalidade compartilhada, use mixins: os Mixins permitem que os componentes compartilhem funcionalidades. O código pode ser reutilizado entre os componentes sem ser duplicado graças às mixinas. Você pode manter seu código DRY (Não se repita) e impedir a duplicação de código utilizando mixins.

CONHECENDO

VUE.JS

Use uma arquitetura baseada em componentes

Uma arquitetura baseada em componentes é uma abordagem de design de software que enfatiza a construção de um sistema a partir de componentes menores e reutilizáveis. No contexto de um projeto Vue.js, uma arquitetura baseada em componentes envolve a divisão da interface do usuário (UI) em componentes menores e reutilizáveis que podem ser facilmente gerenciados e dimensionados.
Aqui estão algumas práticas recomendadas a serem seguidas ao usar uma arquitetura baseada em componentes no seu projeto Vue.js:

1. Divida sua interface do usuário em componentes: identifique elementos da interface do usuário que podem ser reutilizados em diferentes partes do aplicativo e crie componentes individuais para eles. Por exemplo, você pode criar um componente para uma barra de navegação, um componente para uma barra de pesquisa e assim por diante.

2. Mantenha os componentes pequenos e focados: Cada componente deve ter uma finalidade clara e específica. Evite criar componentes muito grandes ou com múltiplas responsabilidades.

3. Use adereços e eventos para se comunicar entre os componentes: use adereços para passar dados dos componentes pai para os componentes filho e eventos para emitir dados dos componentes filho para os componentes pai.

4. Use slots para obter flexibilidade: use slots para permitir que o conteúdo de um componente seja personalizado pelo componente pai.

5. Use uma convenção de nomenclatura para componentes: use uma convenção de nomenclatura que deixe claro o que cada componente faz. Por exemplo, você pode usar um prefixo como “App” para componentes de nível superior e “ Base ” para componentes reutilizáveis.

6. Use uma estrutura de pastas para componentes: Use uma estrutura de pastas lógica para organizar seus componentes. Por exemplo, você pode agrupar componentes relacionados em uma pasta “ componentes ” e organizá-los ainda mais em subpastas, conforme necessário.

7. Use um guia de estilo: use um guia de estilo para garantir consistência no design e desenvolvimento do componente. Um guia de estilo pode definir convenções de nomenclatura, estrutura de componentes e padrões de design que devem ser usados em seu aplicativo.

Use uma convenção de nomenclatura consistente

Uma prática recomendada essencial no desenvolvimento de software é usar um esquema de nomes consistente, pois facilita a leitura e manutenção do código. O uso de um esquema de nomeação consistente para seus arquivos, componentes e variáveis pode facilitar a compreensão e o trabalho do seu código no contexto de um projeto Vue.js.
Os seguintes conselhos ajudarão você a implementar um esquema de nomes consistente em seu projeto Vue.js:

1. Use camelCase para variáveis e funções: No JavaScript, camelCase é a convenção para nomear variáveis e funções. Isso significa que a primeira letra da primeira palavra é minúscula e a primeira letra de cada palavra subsequente é maiúscula. Por exemplo: myVariable, myFunction.

2. Use o PascalCase para componentes: No Vue.js, o PascalCase é a convenção para nomear componentes. Isso significa que a primeira letra de cada palavra é maiúscula e não há separadores entre as palavras. Por exemplo: MyComponent.

3. Use o kebab-case para nomes de arquivos: No Vue.js, o kebab-case é a convenção para nomear arquivos. Isso significa que as palavras são separadas por hífens. Por exemplo: my-component.vue.

4. Use prefixos ou sufixos consistentes: Você pode usar prefixos ou sufixos consistentes para indicar o tipo de arquivo, componente ou variável. Por exemplo, você pode usar um prefixo como “ Base ” para componentes de base usados em seu aplicativo ou um sufixo como “ Serviço ” para classes de serviço que interagem com APIs.

5. Seja descritivo: use nomes descritivos que reflitam com precisão o objetivo do arquivo, componente ou variável. Por exemplo, use nomes como userProfileForm.vue ou getUsersService.js em vez de nomes genéricos como form.vue ou service.js.

Use uma estrutura de pastas

Um grande projeto Vue.js deve ser organizado usando uma estrutura lógica de pastas, pois mantém seu código arrumado e simplifica a navegação e a descoberta do que você precisa.
Aqui estão alguns indicadores para usar uma estrutura de pastas para organizar seu projeto Vue.js:

1. Componentes relacionados ao grupo: crie uma pasta “components” e organize seus componentes em subpastas com base em sua funcionalidade ou finalidade. Por exemplo, você pode ter uma pasta “navbar” para todos os componentes relacionados à barra de navegação e uma pasta “forms” para todos os componentes relacionados ao formulário.

2. Visualizações e componentes separados: crie uma pasta “views” para suas visualizações de nível superior e separe-as dos seus componentes. Isso facilita a distinção entre as visualizações que representam páginas diferentes do seu aplicativo e os componentes usados para criar essas páginas.

3. Crie uma pasta “utils”: Crie uma pasta “utils” para funções ou módulos utilitários que possam ser usados em seu aplicativo, como funções de validação ou módulos de serviço API.

4. Agrupe os módulos Vuex por recurso: se você estiver usando o Vuex para gerenciar seu estado de aplicativo, agrupe seus módulos por recurso ou funcionalidade. Isso facilita a manutenção e depuração do seu código de gerenciamento de estado.

5. Use subpastas para ativos: se você tiver um grande número de ativos, como imagens, fontes ou arquivos CSS, crie subpastas na pasta “ ativos ” para organizá-las por tipo ou finalidade.

6. Use uma convenção de nomenclatura consistente: use uma convenção de nomenclatura consistente para suas pastas e arquivos, como kebab-case para nomes de pastas e PascalCase para nomes de arquivos.

Usar divisão de código

Um método chamado “divisão de código” é usado para dividir um enorme aplicativo JavaScript em partes menores e mais fáceis de gerenciar que podem ser carregadas conforme necessário. Esse método é particularmente útil em grandes projetos do Vue.js, porque o tamanho do aplicativo pode dificultar o gerenciamento e o desempenho.
Os seguintes conselhos ajudarão você a usar a divisão de códigos no seu projeto Vue.js:

1. Use importações dinâmicas: use a sintaxe dinâmica de importação ( ) para carregar componentes ou módulos sob demanda quando forem necessários. Isso permite que você divida seu código em blocos menores que são carregados apenas quando necessários, melhorando o desempenho reduzindo o tempo de carregamento inicial do aplicativo.

2. Use o componente Async do Vue.js: O componente Async do Vue.js permite carregar componentes com preguiça sob demanda, reduzindo o tempo de carga inicial do seu aplicativo. Para usar esse recurso, basta definir seu componente como uma função de assíncrona que retorna a definição do componente.

3. Use o router Vue.js: O roteador Vue.js permite definir rotas para sua aplicação e carregar com preguiça os componentes associados a essas rotas. Isso significa que os componentes são carregados apenas quando a rota é acessada, melhorando o desempenho do seu aplicativo.

4. Use o webpack ou outros empacotadores: o Webpack é um empacotador popular que inclui suporte interno para divisão de código. Ao configurar o webpack para dividir seu código em pedaços menores, você pode melhorar o desempenho e a manutenção do seu projeto Vue.js.

5. Analise seu aplicativo: use ferramentas como o analyzer webpack-bundle para analisar seu aplicativo e identificar áreas em que a divisão de código pode ser usada para melhorar o desempenho.

Use ferramentas de ligação e formatação

Ao aplicar o estilo e a uniformidade do código, as ferramentas de vinculação e formatação são cruciais para o gerenciamento de um grande projeto Vue.js. Para usar as ferramentas de ligação e formatação no seu projeto Vue.js, siga estas dicas:

1. Use ESLint: O ESLint é uma ferramenta de vinculação popular para JavaScript que pode ajudá-lo a identificar e corrigir erros e aplicar o estilo e a consistência do código. Você pode configurar o ESLint para trabalhar com o Vue.js instalando o plug-in eslint-plugin-vue.

2. Use Prettier: Prettier é uma ferramenta de formatação de código que pode ajudá-lo a aplicar um estilo de código consistente em todo o projeto Vue.js. Prettier pode ser usado ao lado do ESLint para formatar seu código automaticamente enquanto você o escreve.

3. Configure suas regras de vinculação e formatação: configure suas regras ESLint e Prettier para aplicar o estilo e a consistência do código que você deseja obter no seu projeto Vue.js. Você pode configurar suas regras para corresponder aos padrões de codificação de sua equipe ou às melhores práticas do setor.

4. Integre a ligação e a formatação ao seu processo de desenvolvimento: integre suas ferramentas de ligação e formatação ao seu processo de desenvolvimento, adicionando-as ao seu editor de código ou criando pipeline. Isso pode ajudá-lo a detectar erros e reforçar a consistência antes que o código seja comprometido com o sistema de controle de versão.

5. Use um gancho de pré-confirmação: use um gancho de pré-confirmação para executar suas ferramentas de ligação e formatação automaticamente antes que o código seja comprometido com o sistema de controle de versão. Isso pode ajudar a garantir que seu código atenda aos padrões de sua equipe e evite que erros sejam cometidos no seu repositório.

CONHECENDO

VUE.JS

Documente seu código

O primeiro passo para planejar um grande projeto Vue.js é documentar seu código. Ajuda na compreensão da equipe sobre a funcionalidade do código. Aqui estão algumas diretrizes para escrever seu código Vue.js:

1. Use JSDoc: O JSDoc é uma ferramenta de documentação popular para JavaScript que permite adicionar comentários ao seu código formatados como tags. Essas tags podem ser usadas para documentar a finalidade, parâmetros e valores de retorno das funções, entre outras coisas.

2. Documente seus componentes: Ao criar componentes, documente seus adereços, eventos e slots. Isso pode ajudar outros desenvolvedores a entender como usar seus componentes e como eles se encaixam na estrutura geral do seu projeto Vue.js.

3. Documente sua lógica Vuex: se você estiver usando o Vuex para gerenciar o estado do seu projeto Vue.js, documente sua lógica adicionando comentários que explicam o objetivo de cada módulo, estado, mutação, e ação.
4. Escreva comentários claros e concisos: Escreva comentários claros e concisos. Use linguagem simples e evite o jargão técnico o máximo possível. Além disso, verifique se seus comentários estão atualizados e precisos.

5. Use um estilo consistente: use um estilo consistente para seus comentários em seu projeto Vue.js. Isso pode facilitar a leitura e a compreensão da base de código, especialmente se você estiver trabalhando com vários desenvolvedores.

6. Use arquivos README: Escreva arquivos README para cada módulo, componente ou recurso do seu projeto Vue.js. Esses arquivos podem fornecer uma visão geral do que o módulo ou recurso faz e como usá-lo.

Referencia: @nile.bites