Pinia

**# Pinia基本概念**

>面试题：Pinia 相比 Vuex 有什么样的优点？为什么现在官方推荐使用 Pinia ？

Pinia，是一个 Vue 阵营的新的状态管理库，现在 Vue 官方已经推荐使用 Pinia 来代替 Vuex，或者你可以把 Pinia 看作是 Vuex 的最新的版本。

- Pinia 的基本介绍

- Pinia 优势

**## Pinia 的基本介绍**

Pinia 是一个西班牙语的单词，表示“菠萝”的意思。因为菠萝是由一小块一小块组成的，这个和 Pinia 的思想就非常的吻合，在 Pinia 中，每个 Store 仓库都是单独的扁平化的存在的。

Pinia 是由 Vue 官方团队中的一个成员开发的，最早是在 2019 年 11 左右作为一项实验性工作所提出的，当时的目的是将组合 API 融入到 Vuex 中，探索新版本的 Vuex 应有的形态，随着探索的进行，最终发现 Pinia 已经实现了 Vuex5 大部分的提案，因此 Pinia 就作为了最新版本的 Vuex，但是为了尊重作者本人，名字保持不变，仍然叫做 Pinia。

相比 Vuex，Pinia 的 API 更少而且更简单，还支持组合式 API，还可以和 Typescript 一起使用来做类型的推断。

pinia 官网：https://pinia.vuejs.org/

<img src="https://xiejie-typora.oss-cn-chengdu.aliyuncs.com/2023-03-21-093840.png" alt="image-20230321173840739" style="zoom:50%;" />

**## Pinia 优势**

1. 在 Pinia 中，已经不存在 mutations，只有 state、getters、actions

```js

import { defineStore } from 'pinia'

export const useCounterStore = defineStore('counter',{

  state: () => ({

    count: 0

  }),

  getters: {

    doubleCount: state => state.count \* 2

  },

  actions: {

    increment() {

      this.count++

    },

  }

})

```

在上面的代码中，我们创建了一个仓库，该仓库中提供三个选项，分别是 state、getters 以及 actions。

2. actions 里面支持同步和异步来修改 store，相当于将之前 Vuex 中的 mutation 和 action 合并了

```js

import { defineStore } from 'pinia'

export const useCounterStore = defineStore({

  // ...

  actions: {

    // 同步的修改仓库状态

    increment() {

      this.count++

    },

    decrement() {

      this.count--

    },

    // 异步的修改仓库状态

    async incrementAsync() {

      await new Promise(resolve => setTimeout(resolve, 1000))

      this.increment()

    },

    async decrementAsync() {

      await new Promise(resolve => setTimeout(resolve, 1000))

      this.decrement()

    }

  }

})

```

3. 可以和 TypeScript 一起使用，以此来获得类型推断的支持

```js

import { defineStore } from 'pinia'

// 这里定义了一个名为 Todo 的接口

interface Todo {

  id: number;

  text: string;

  done: boolean;

}

export const useTodoStore = defineStore({

  id: 'todo',

  state: () => ({

    todos: [] as Todo[],

  }),

  getters: {

    completedTodos: state => state.todos.filter(todo => todo.done),

  },

  actions: {

    // text 指定了是 string 类型

    addTodoItem(text: string) {

      const id = state.todos.length + 1

      const newTodo = { id, text, done: false }

      state.todos.push(newTodo)

    },

    // todo 指定了是 Todo 类型

    toggleTodoItem(todo: Todo) {

      todo.done = !todo.done

    },

    async fetchTodos() {

      const response = await fetch('https://jsonplaceholder.typicode.com/todos')

      const todos = await response.json() as Todo[]

      state.todos = todos

    },

  },

})

```

4. 关于 Store 仓库，每一个 Store 仓库都是独立的扁平化的存在的，不再像 Vuex 里面是通过 modules 嵌套

5. 支持插件扩展，可以通过插件（函数）来扩展仓库的功能，为仓库添加全局属性或者全局方法

```js

// ...

// 这里定义了一个名为 localStoragePlugin 的插件，本质上就是一个函数

const localStoragePlugin = (context: PiniaPluginContext) => {

  const key = 'my-app-state'

  // 从 localStorage 中恢复状态

  context.state = localStorage.getItem(key) || context.state

  // 监听 state 变化，将变化保存到 localStorage

  context.subscribe((mutation) => {

    localStorage.setItem(key, context.state)

  })

}

// ...

// 创建 Pinia 实例，并注册 localStoragePlugin 插件

const pinia = createPinia()

pinia.use(localStoragePlugin)

```

6. 更加轻量，压缩之后体积只有 1kb 左右，基本上可以忽略这个库的存在

**## 真题解答**

> 题目：Pinia 相比 Vuex 有什么样的优点？为什么现在官方推荐使用 Pinia ？

>

> 参考答案：

>

> Pinia 是由 Vue.js 团队成员开发的下一代状态管理仓库，相比 Vuex 3.x/4.x，Pinia 可以看作是 Vuex5 版本。

>

> Pinia 具有如下的优势：

>

> - mutations 不复存在。只有 state 、getters 、actions。

>

> - actions 中支持同步和异步方法修改 state 状态。

>

> - 与 TypeScript 一起使用具有可靠的类型推断支持。

>

> - 不再有模块嵌套，只有 Store 的概念，Store 之间可以相互调用。

>

> - 支持插件扩展，可以非常方便实现本地存储等功能。

>

> - 更加轻量，压缩后体积只有 1kb 左右。