# State Management in Vue

## Introduction to State Management in Vue+Pinia

Yoosuf, 26-03-2025

# Agenda

- Introduction to State Management in Vue
- Vue's Built-in State Management
- The Need for Dedicated State Management
- Introduction to Pinia
- Key Features of Pinia
- Setting Up Pinia
- Using Pinia in Components
- Extending Pinia with Plugins
- Questions and Answers

# Introduction to State Management in Vue

- Vue’s reactivity system
- Importance of state management
- Common state management patterns

# Vue's Built-in State Management

- Local component state (data, props, computed properties)
- Provide/Inject for dependency injection
- Reactive global state using reactive() and ref()

# The Need for Dedicated State Management

- Challenges with prop drilling
- Managing complex state across multiple components
- Ensuring reactivity and performance optimization

# Introduction to Pinia

- Official state management library for Vue
- Designed as a modern replacement for Vuex
- Simplicity, performance, and TypeScript support

# Key Features of Pinia

- Modular store structure
- Reactivity using Vue's Composition API
- Support for SSR (Server-Side Rendering)
- Devtools integration

# Setting Up Pinia

1.Install Pinia:

```
npm install pinia -S
```

2. Create a simple Pinia store:

```ts
import { defineStore } from 'pinia';

interface Todo {
  id: number;
  text: string;
}

export const useTodoStore = defineStore('todo', {
  state: () => ({ todos: [] as Todo[] }),
  actions: {
    addTodo(text: string) {
      this.todos.push({ id: Date.now(), text });
    },
    removeTodo(id: number) {
      this.todos = this.todos.filter(todo => todo.id !== id);
    },
  },
});
```

3. Register Pinia in main.js:

```
import { createApp } from 'vue';
import { createPinia } from 'pinia';
import App from './App.vue';

const app = createApp(App);
app.use(createPinia());
app.mount('#app');
```

# Using Pinia in Components

- Importing and accessing the store:

```
import { useTodoStore } from '@/stores/todo';

const todoStore = useTodoStore();
todoStore.addTodo('Learn Pinia with TypeScript');
console.log(todoStore.todos);
```

# Extending Pinia with Plugins

- Pinia supports plugins to extend functionality
- Example of a plugin for local storage persistence:

```
import { defineStore } from 'pinia';

export const useUserStore = defineStore('user', {
  state: () => ({ name: '', email: '' }),
  persist: true,
});
```

- There are already predefined plugins available at https://pinia.vuejs.org/core-concepts/plugins.html

# Conclusion

- Pinia simplifies Vue state management
- Supports modular architecture and better performance
- Ideal for Vue 3 applications
- Encourages maintainable and scalable code structures