

Vue.js Vuex

Advanced State Management for Vue.js

Nested Components

<App>

<Topbar />

<Sidebar />

<Main>

<Settings>

<PersonalSettings>

Sidebar Toggle: on/off

</PersonalSettings>

</Settings>

</Main>

</App>

Installation

Vue CLI

`vue create folder`

Select vuex

Install Vuex on existing project

```
npm install vuex
```

```
#create new file to store vuex data
```

```
mkdir -p store
```

```
touch store/index.js
```

main.js

```
import Vue from 'vue'
import Vuex from 'vuex'

Vue.use(Vuex)

const store = new Vuex.Store()

new Vue({
  store
}).$mount('#app')
```

main.js

```
// ...  
const store = new Vuex.Store({  
  state: {  
    count: 0  
  },  
  mutations: {  
    increment (state) {  
      state.count++  
    }  
  }  
})
```



States

States - manually assign to data

```
<template>
  <div>{{ count }}</div>
</template>

<script>
  export default {
    // ...
    mounted() {
      this.count = this.$store.state.count
    }
  }
</script>
```

States with computed

```
<template>
  <div>{{ count }}</div>
</template>

<script>
  export default {
    // ...
    computed: {
      count() {
        return this.$store.state.count
      }
    }
  }
</script>
```

Mutations

Mutations

```
<template>
  <button @click="increase">Increase</button>
</template>

<script>
  export default {
    // ...
    methods: {
      increase() {
        this.$store.commit('increment')
      }
    }
  }
</script>
```



Mutations Must Be Synchronous

main.js

```
// ...  
const store = new Vuex.Store({  
  state: {  
    count: 0  
  },  
  mutations: {  
    increment (state, value = 1) {  
      state.count += value  
    }  
  }  
})
```

Mutations with value

```
<template>
  <button @click="increase">Increase +3</button>
</template>

<script>
  export default {
    // ...
    methods: {
      increase() {
        this.$store.commit('increment', 3)
      }
    }
  }
</script>
```

Define an Action

```
const store = new Vuex.Store({  
  actions: {  
    increment(context) {  
      context.commit('increment')  
    }  
  }  
})
```


Define an Action with destructured object

```
const store = new Vuex.Store({  
  actions: {  
    increment({ commit, state }) {  
      commit('increment')  
      console.log(state.count)  
    }  
  }  
})
```

Actions

```
<template>
  <button @click="increase">Increase +3</button>
</template>

<script>
  export default {
    // ...
    methods: {
      increase() {
        this.$store.dispatch('increment')
      }
    }
  }
</script>
```

Define an Action with destructured object

```
const store = new Vuex.Store({  
  actions: {  
    increment({ commit, state }, value) {  
      commit('increment', value)  
      console.log(state.count)  
    }  
  }  
})
```

Actions with value

```
<template>
  <button @click="increase">Increase +3</button>
</template>

<script>
  export default {
    // ...
    methods: {
      increase() {
        this.$store.dispatch('increment', 8)
      }
    }
  }
</script>
```



Actions Can Be Asynchronous



Async Actions

```
const store = new Vuex.Store({
  actions: {
    async send({ commit }) {
      let res = await fetch('https://httpbin.org/get')
      let json = await res.json()
      commit('data', json)
    }
  }
})
```

Getters

Getters

```
const store = new Vuex.Store({  
  getters: {  
    count: state => {  
      return state.count  
    }  
  }  
})
```


Getters

```
<script>
  computed: {
    count() {
      return this.$store.getters.count
    }
  }
</script>
```

Modules

Modules

```
const store = new Vuex.Store({  
  moduleA: {  
    namespaced: true,  
    state,  
    mutations,  
    getters,  
    actions,  
  },  
  moduleB: {  
    namespaced: true,  
    state,  
    mutations,  
    getters,  
    actions,  
  }  
})
```

Modules

```
<script>  
  this.$store.moduleA.state.count  
</script>
```

Vuex Plugins

Vuex Plugins

1. vuex-persistedstate
2. vuex-pathify

Q & A



<https://vuex.vuejs.org>



Thank you