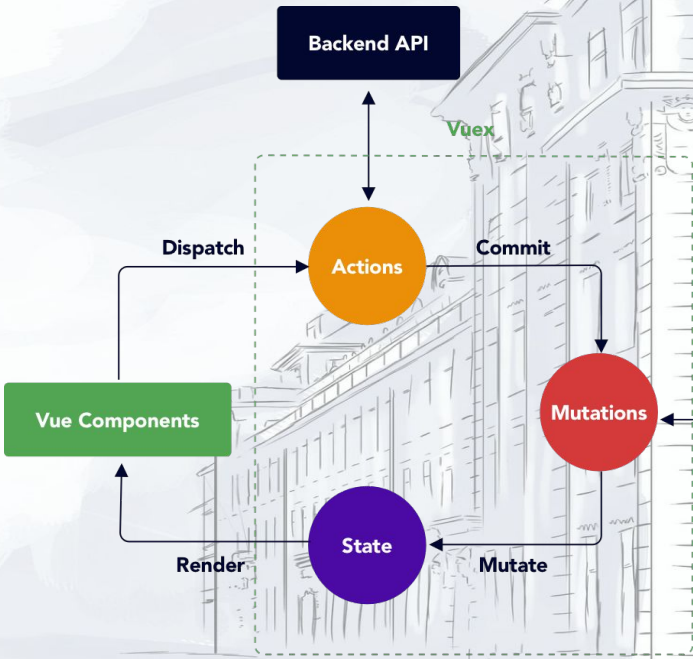
Vuex - Evil Singleton

Was man über Vuex wissen sollte

Die schöne Idee



Die hässliche Realität

State ist ein Singleton

- code smell
- eigenen Lifecycle
- starke Kopplung
- status behaftet

Problem

- Jeder darf das Singleton verändern
- Das verbietet kein Sprachfeature
- Das prüft kein Compiler
- Das prüft kein Linter

https://github.com/jwausle/vuex-evil-singleton

Konkret

```
1   import Vue from 'vue';
2   import Vuex from 'vuex';
3
4   Vue.use(Vuex);
5
6   export default new Vuex.Store({
7     state: {
8       foo: { bar: 'stored value' },
9     },
10    getters: { },
11    mutations: { },
12    actions: { },
13  });
```

```
1   <template>
2     <div class="home">
3       <h1>Outer 'foo.bar': {{getFoo.bar}} </h1>
4     </div>
5   </template>
6
7   <script lang="ts">
8   import Store from '../store'
9   import { Component, Vue } from 'vue-property-de
10
11  @Component
12  export default class Home extends Vue {
13    get getFoo() { return Store.state.foo; }
14  }
15  </script>
```

https://github.com/jwausle/vuex-evil-singleton

Wo ist jetzt das Problem?

```
11  @Component
12  export default class FooFromStore extends Vue {
13      private foo:any = { bar: '' };
14
15      private mounted() { this.foo = Store.state.foo; }
16
17      get fooBar() { return this.foo.bar; }
18
19      set fooBar(newFooBar:string){
20          this.foo.bar = newFooBar;
21
22          const localFooFreshFromStore = Store.state.foo;
23          if( localFooFreshFromStore.bar === this.foo.bar) {
24              console.log('>>> Who the fuck changed the store value.');
25          }
26      }
27  }
28  </script>
```

https://github.com/jwausle/vuex-evil-singleton

2 Lösungsideen

Use getters they return copies

```
1  import Vue from 'vue';
2  import Vuex from 'vuex';
3
4  Vue.use(Vuex);
5
6  export default new Vuex.Store({
7    state: {
8      foo: { bar: 'stored value' },
9    },
10   getters: {
11     fooCopy(state) { return Object.assign({}, state.foo); },
12   },
13   mutations: {
14     setFoo(state, foo) { state.foo = foo; },
15   },
16   actions: { },
17 });
```

Mutate only state sub parts

```
1  import Vue from 'vue';
2  import Vuex from 'vuex';
3
4  Vue.use(Vuex);
5
6  export default new Vuex.Store({
7    state: {
8      foo: { bar: 'stored value' },
9    },
10   getters: {
11   },
12   mutations: {
13     setFooBar(state, bar) { state.foo.bar = bar; },
14   },
15   actions: { },
16 });
```

https://github.com/jwausle/vuex-evil-singleton

Fertsch

https://github.com/jwausle/vuex-evil-singleton