

Glossary

Getters - Functions used to return values from the store. Use these when computations need to be performed on the state value before they are passed to the caller. (You can also access the state directly.)

Mutations - Functions that commit changes to the state, and can process the values being passed before saving them. (You can also modify the state directly.)

Actions - Similar to mutations, except they commit data typically using during asynchronous tasks. They're optional, but you should get into the habit of using these whenever mutations are present. Vuex provides many more features outside of the scope of this quick reference. Consult the manual for more information!

<https://vuex.vuejs.org/guide/>

Vuex 2.1 adds namespacing for modules! <https://github.com/vuejs/vuex/releases/tag/v2.1.0>

The Store Object

```
import Vue from 'vue';
import Vuex from 'vuex';
Vue.use(Vuex);
export const store = new Vuex.Store({
  state: {
    my_property: 47
  },
  getters: {
    getMyProperty: state => {
      return state.my_property;
    }
  },
  mutations: {
    setMyProperty: (state, payload) => {
      state.my_property = payload;
    }
  },
  actions: {
    doChangeMyProperty: (context, payload) => {
      context.commit('setMyProperty', payload);
    }
  }
});
```

The Store Object (cont)

```
> doAsyncChangeMyProperty: ({ commit }, payload) => {
  // Using a timeout, but any async operation can go here
  setTimeout(function () {
    commit('setMyProperty', payload);
  }, 1000);
}
});
```

The *getter*, *mutations*, and *actions* properties are optional.

Modules

```
const moduleA = {
  state: { ... },
  mutations: { ... },
  actions: { ... },
  getters: { ... }
}

const moduleB = {
  state: { ... },
  mutations: { ... },
  actions: { ... }
}

const store = new Vuex.Store({
  modules: {
    a: moduleA,
    b: moduleB
  }
})

store.state.a // moduleA's state
store.state.b // moduleB's state
```

From <https://vuex.vuejs.org/guide/modules.html>



By **fortyseven**

cheatography.com/fortyseven/

Published 18th June, 2018.

Last updated 18th June, 2018.

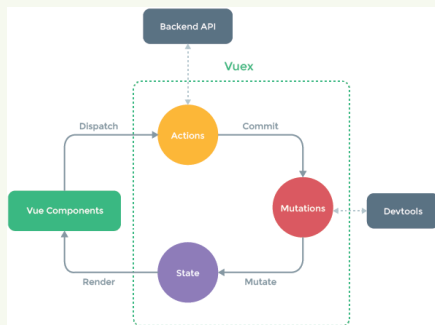
Page 1 of 3.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>

The Vuex Cycle



Using Mutations (cont)

```

</template>
<script>
import { mapMutations } from 'vuex';
export default {
  methods: {
    ...mapMutations([
      'setMyProperty'
    ]),
    // Or without helper...
    setThatProp( value ) {
      this.$store.commit( 'setMyProperty', value );
    }
  }
}
</script>

```

Using Getters

```

<template>
  <div v>
    <div v>Value = {{ getMyProperty }}
  </div v>
    <div v>Value = {{ getStoreProp }}
  </div v>
  </div>
</template>
<script>
import { mapGetters } from 'vuex';
export default {
  computed: {
    ...mapGetters([
      'getMyProperty'
    ]),
    // or, without helper...
    getStoreProp: () => {
      return
this.$store.getters.getMyProperty;
    }
  }
}
</script>

```

Using Actions

```

<template>
  <div v>
    <button @click ="doChangeMyProperty(5)">Change me to 5</button>
  </div v>
</template>
<script>
import { mapActions } from 'vuex';
export default {
  methods: {
    ...mapActions([
      'doAsyncChangeMyProperty',
    ]),
    // Or without helper...
    doThatThing( value ) {
      this.$store.dispatch( 'doAsyncChangeMyProperty',
value);
    }
  }
}
</script>

```

Using Mutations

```

<template>
  <div v>
    <button @click ="setMyProperty( 22 )" >Set to 22</button>
    <button @click ="setThatProperty( 0 )" >Reset to 0</button>
  </div>

```



Two-Way Binding

```
<template>
  <div v=
    <input type="text" v-model="my_property" />
  </div>
</template>
<script>
  export default {
    computed: {
      my_property: {
        get() {
          return
this.$state.get('my_property');
        },
        set(value) {
          this.$state.set('my_property', value);
        }
      }
    }
  }
</script>
```

This usage of computed properties with a getter/setter property is rarely used!



By **fortyseven**
cheatography.com/fortyseven/

Published 18th June, 2018.
Last updated 18th June, 2018.
Page 3 of 3.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>