Managing Data State with Vuex

John Papa
DEVELOPER ADVOCATE

@john_papa www.johnpapa.net

State Management with Vuex

Managing state in the browser with your Vue app

https://vuex.vuejs.org/

How do we ...

Add state management to our app to centralize the data?

Define what goes into state ?

Define how to fetch and change the state?

Communicate with the state from components?

Demo

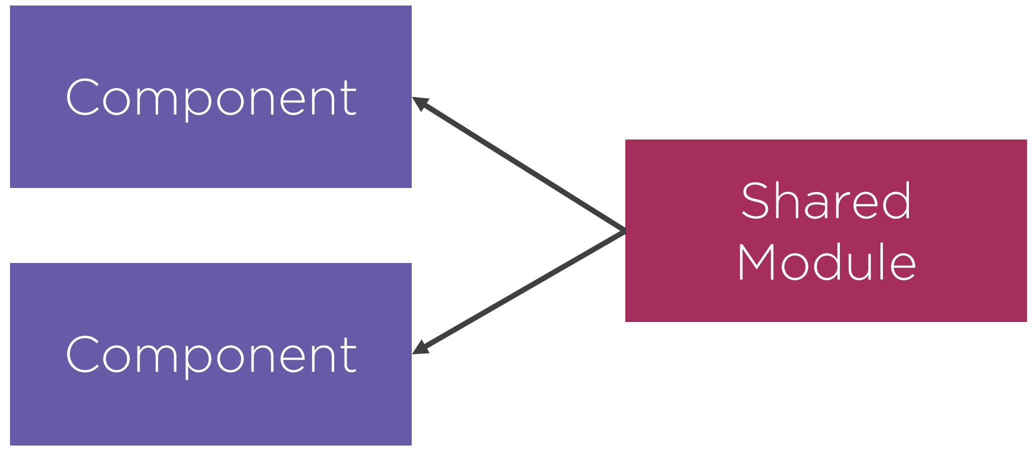
No Central Store

Centralizing the Data Flow with Vuex

Types of Shared State

Entity	Heroes, Villains, Customers, Orders
Session	User info, preferences, app settings, UX state, router config
Local	Class properties are still perfectly good for a class component

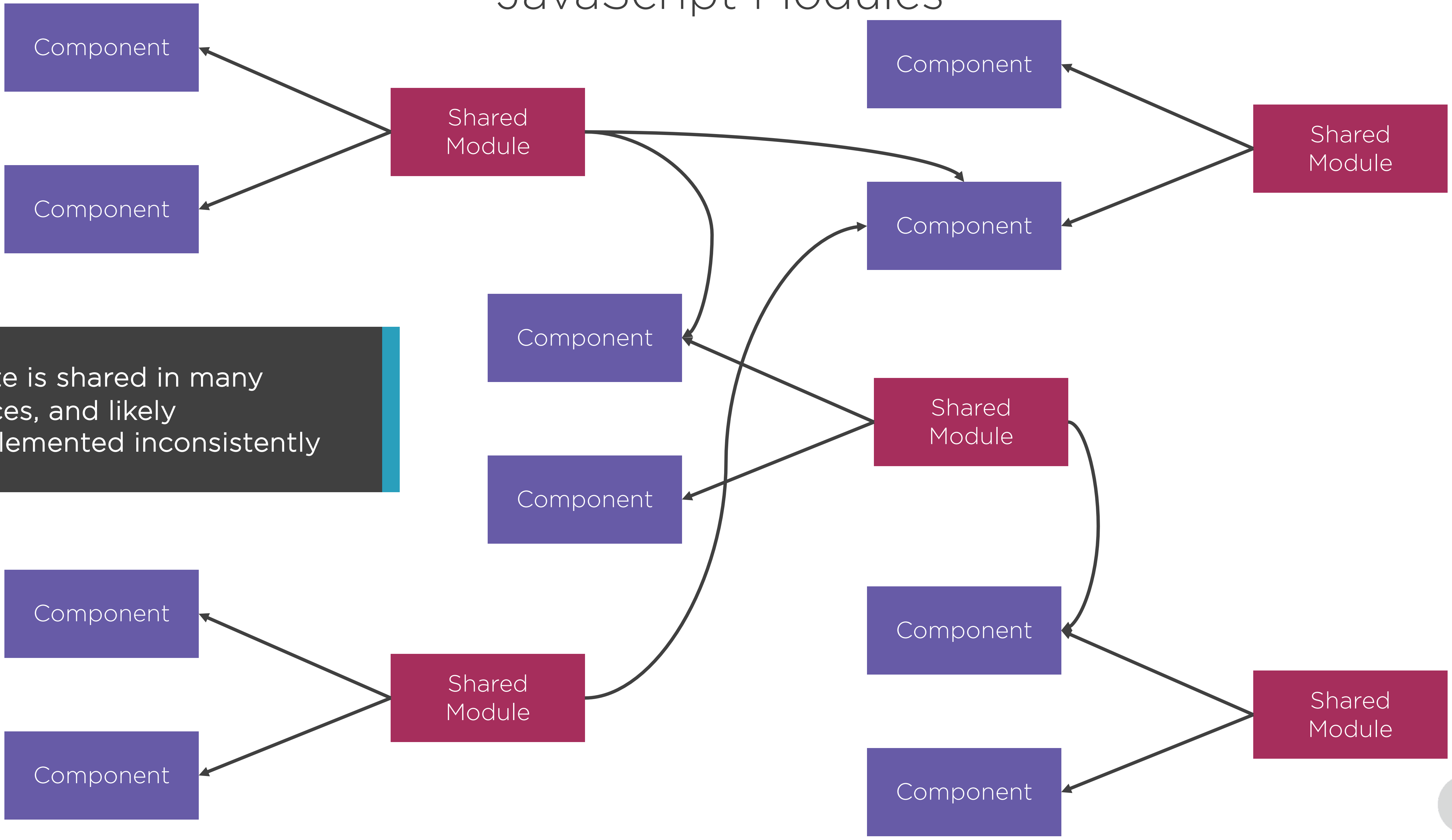
JavaScript Modules



Simplest approach

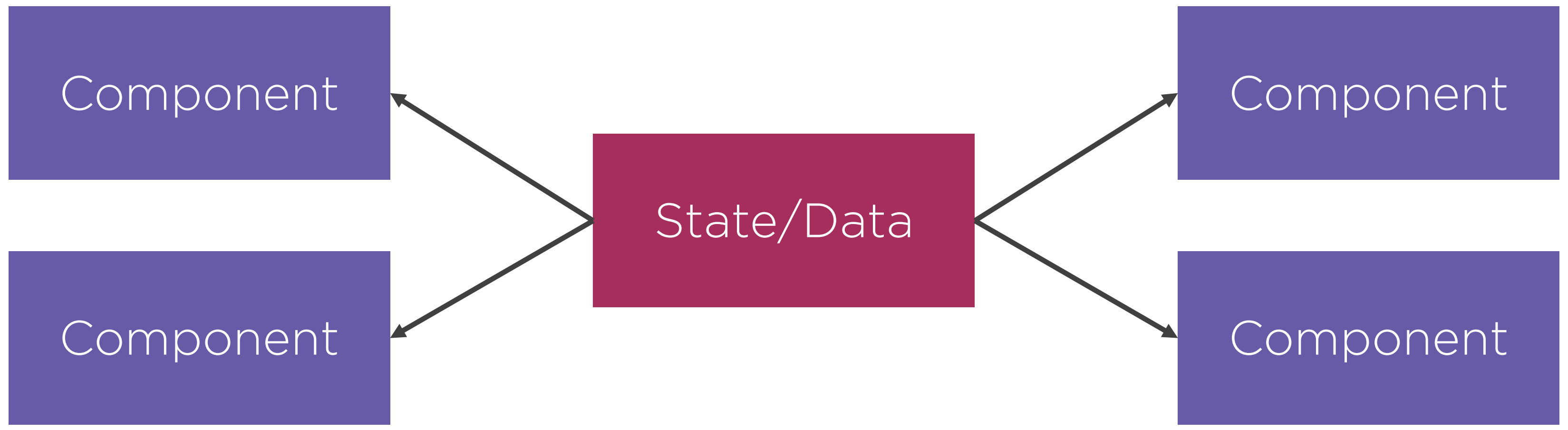
Inject services where they are needed

JavaScript Modules



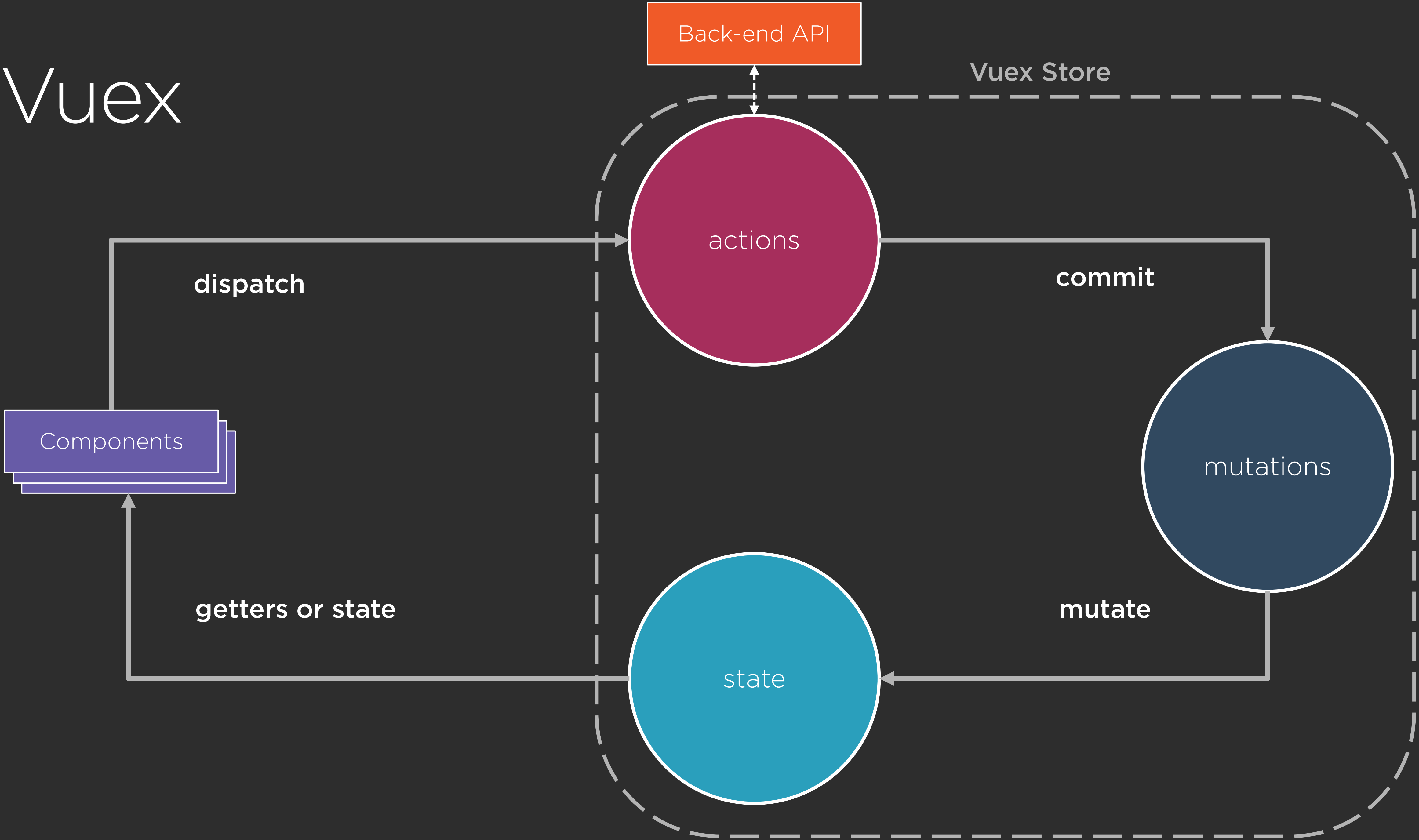
State is shared in many places, and likely implemented inconsistently

Let’s Get Back to a Single Source of Data

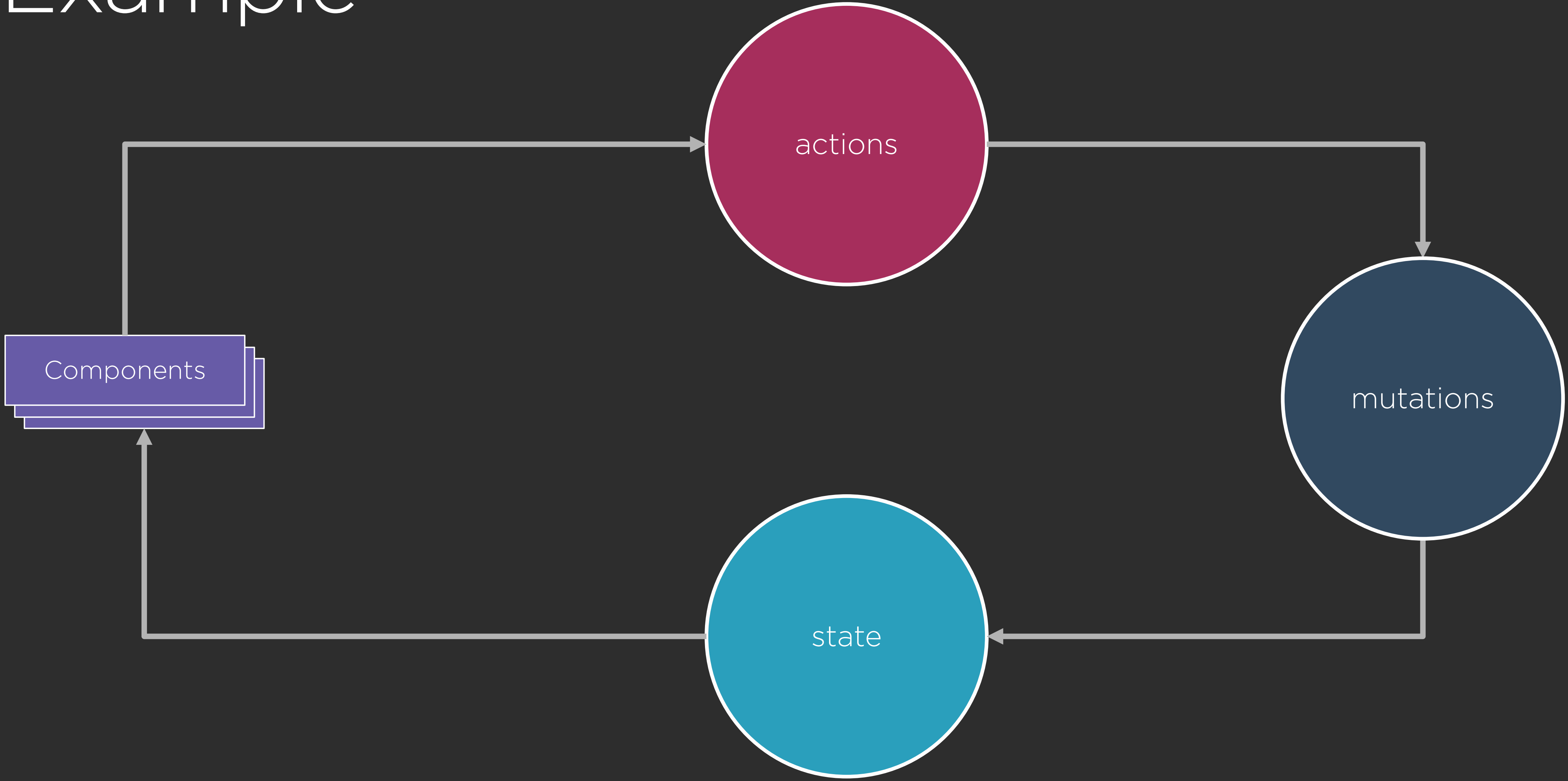


Vuex helps create a single source of truth for your data

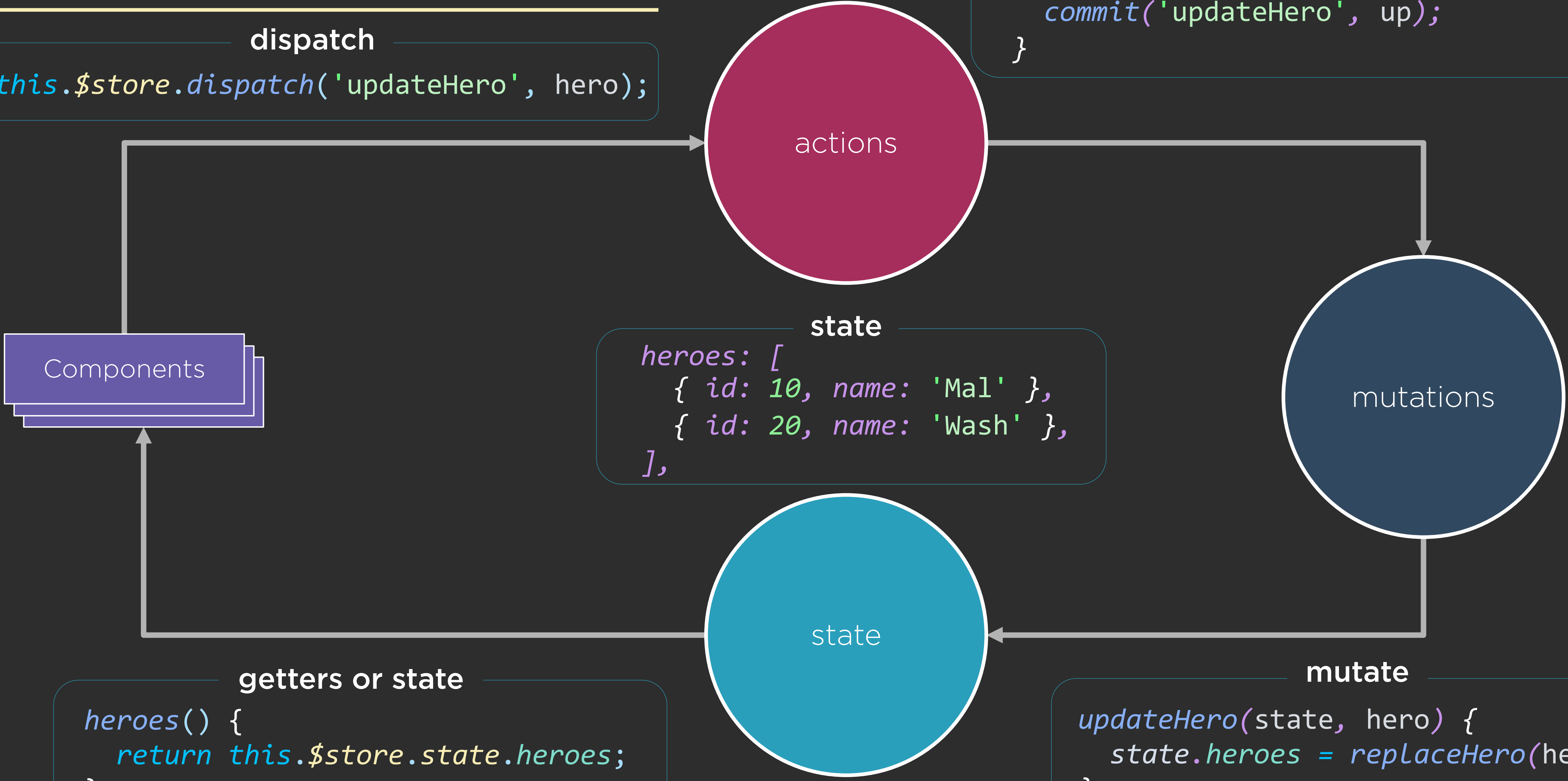
Vuex


Back-end API
Vuex Store
actions
dispatch
commit
Components
mutations
getters or state
mutate
state

Example



Component gets the updated heroes



dispatch

```
this.$store.dispatch('updateHero', hero);
```

commit

```
async updateHero({ commit }, hero) {
  const up = await svc.updateHero(hero);
  commit('updateHero', up);
}
```

state

```
heroes: [
  { id: 10, name: 'Mal' },
  { id: 20, name: 'Wash' },
],
```

mutate

```
updateHero(state, hero) {
  state.heroes = replaceHero(hero);
},
```

getters or state

```
heroes() {
  return this.$store.state.heroes;
},
```

Demo

Adding Vuex to Your App

Adding Vuex to Your Vue App

1 Add Vuex to your Vue app

2 Customize your store

3 Tell your app about the store

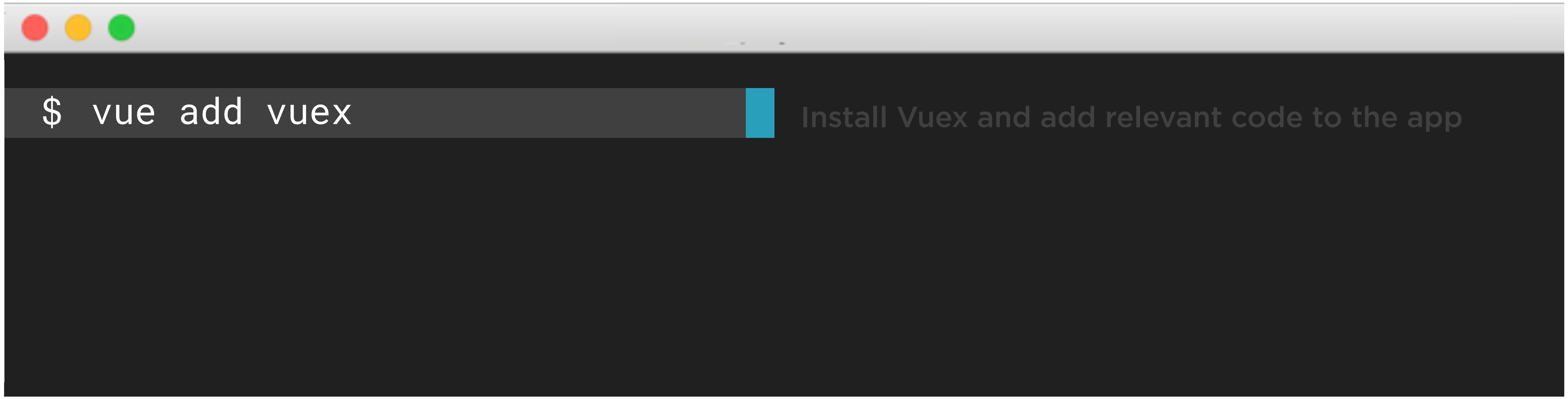
4 Communicate with the store from your components

Installing Vuex

Easiest way is to use the Vue CLI's vue add syntax

https://vuex.vuejs.org/

https://cli.vuejs.org/



```
$ vue add vuex
```

Install Vuex and add relevant code to the app

Adding Vuex

Import the store definition

Add the store to the app

The Vue CLI does the work for you

```
vue add vuex
```

main.js

```
import Vue from 'vue';

import App from '@/app.vue';

import store from './store';

new Vue({

  store,

  render: h => h(App)

}).$mount('#app');
```

store.js

```
import Vue from 'vue';

import Vuex from 'vuex';


Vue.use(Vuex);


export default new Vuex.Store({

  state: {},

  mutations: {},

  actions: {},

  getters: {},

});
```

Connect Vuex to the app

Creates the Vuex store for the app's state data

Defines the state, mutations, actions and getters for your store
(we'll fill these in together)

Demo

Describing Your State

State

The data you want in your store

Always start with an initial state

store.js

```
state: {
  heroes: [],
  villains: [],
  user: {}
},
```

Demo

Fetching State

store.js

```
mutations: {
  addHero(state, payload) {
    state.heroes.push(payload);
  }
}
```

Mutations

Synchronous operations to modify your state

Actions

Functions

Can commit one or more mutations

Can call other actions

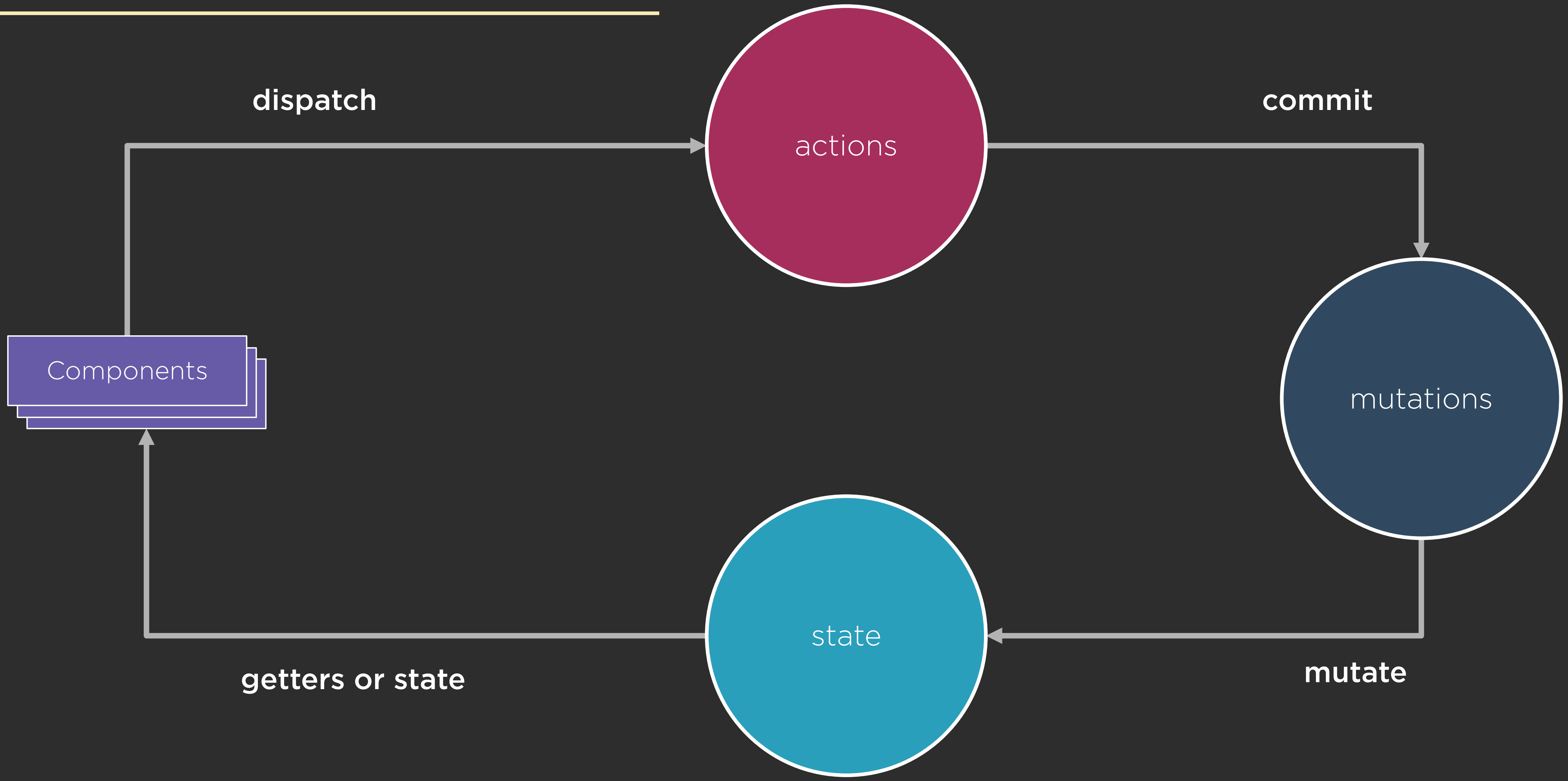
Can be sync or async

Call these from your components

store.js

```
const actions = {
  async addHero({ commit }, hero) {
    const h = await svc.addHero(hero);
    commit(ADD_HERO, h);
  },
};
```

Component gets the heroes



Demo

Getting Specific State

store.js

```
getters: {
  damageTotal: ({ bill }) => {
    let total = 0;
    for (let item of bill) {
      total += item.cost * item.qty;
    }
    return total;
  },
},
```

Getters

Make it easier to access parts of state

Can summarize or get specific state

Demo

Strict Mode

Demo

Mutating State

Demo

Actions

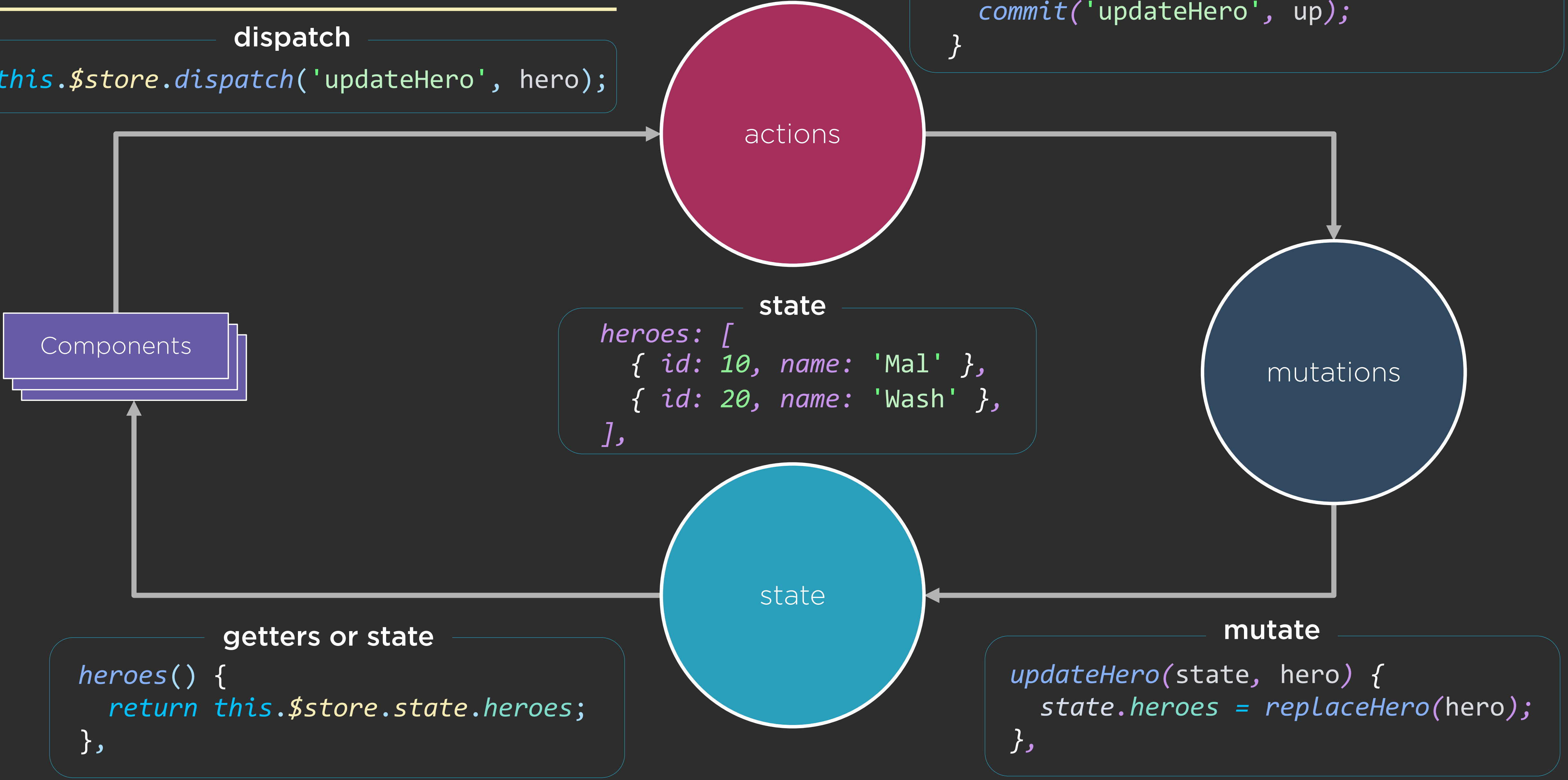
Demo

Communicate Between Components and the Store

Demo

A Tour of Heroes and Villains

Component gets the updated heroes



dispatch

```
this.$store.dispatch('updateHero', hero);
```

commit

```
async updateHero({ commit }, hero) {
  const up = await svc.updateHero(hero);
  commit('updateHero', up);
}
```

state

```
heroes: [
  { id: 10, name: 'Mal' },
  { id: 20, name: 'Wash' },
],
```

getters or state

```
heroes() {
  return this.$store.state.heroes;
},
```

mutate

```
updateHero(state, hero) {
  state.heroes = replaceHero(hero);
},
```

What we Learned

Add Vuex with the Vue CLI

How to create state

How to dispatch to actions that commit mutations to the state

Using the mappers to communicate from components to the store

Summary

When state is shared , Vuex helps

Mapper functions reduce code

Vuex docs https://vuex.vuejs.org/