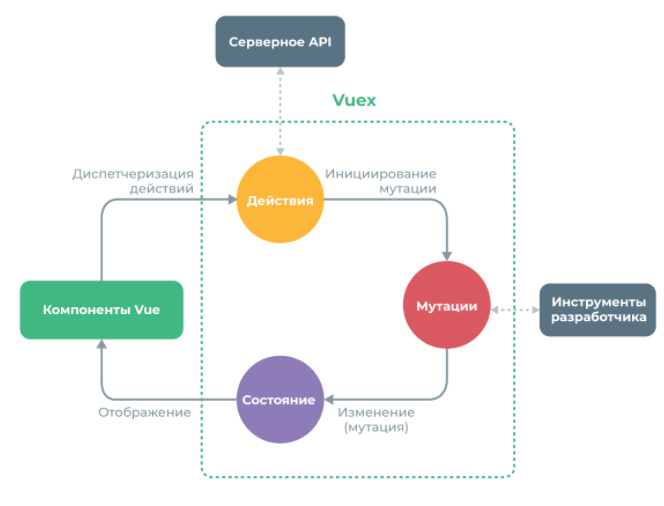
**Что такое Vuex?**



# Components -> данныйды Actions-қа жібереді

      this.$store.dispatch("increment", this.number);

# Actions-данныйды қабылдап Mutation - ға жіберді

  actions: {

    increment({ commit }, payload) {

      commit("increment", payload);

    },

  },

# Mutations ол данныйды функцияға салады, өзгертеді

# оны State - те глобаль переменныйға теңестіреді

state: {

    arr: [],

    count: 0,

  },

mutations: {

    increment(state, payload) {

      state.arr = payload;

    },

  },

# Getters - арқылы -> Components -тен көрсетеді

# (*State - тен сразу шығару кей жағдайда қате береді)*

  getters: {

    COUNT: (state) => state.arr,

  },

# *Шығаратын беттен импорт арқылы шақыру керек*

import { mapGetters } from "vuex";

  computed: {

    ...mapGetters(["COUNT"]),

  },

  <div>

    {{ COUNT  }}

  </div>

**Действия (Actions)** - данный қабылдау, серверден данный алады, ассинхронды байланыс қыла алады, мутатциядағы событяны шақыра алады, (шығарады..көрсете алады)

**Мутации (Mutations)** - данныйларды өзгерту, события, синхронды болады,

**Состояние (State)** - Data сияқты данныйлар жазылады, барлық компонентке доступ болатын переменныйлар ,

осы данный дағы заттарды Мутация өзгертеді  
**Геттеры (Getters)** - көрсету алдында filter қылу, ұзындығын, санын анықтап көрсетуге болады ,

**Модули (Modules)** - үлкен проекттерде түсінікті болу үшін бөліп тастау

Данныйлар **Actions**-қа жіберіледі

**Mutations** -та өңделеді

**Getters** -те қайта (көрсетіледі) шақырылады

----------------------------------

**#State**

## Getting Vuex State into Vue Components

`  state: {

  numberState: 10,

    posts: [1, 4, 34, 7, 3, 6],

  },

  console.log(this.$store.state.post.posts);

## The mapState Helper ???

import { mapState } from 'vuex'

computed: mapState({

// arrow functions can make the code very succinct!

count: state => state.count,

// passing the string value 'count' is same as `state => state.count`

countAlias: 'count',

// to access local state with `this`, a normal function must be used

countPlusLocalState (state) {

return state.count + this.localCount

}

})

Егер а- деген жеке модуль ішінде тұрса істемей қалады, оны істету үшін төмендегі мысалға қарау керек

import {  mapState } from "vuex";

computed: {

    ...mapState({

      count: (state) => state.a.count,

    }),

  },

    <h1>{{ count   }}</h1>

----------------------------------

**#Getting**

## Property-Style Access ???

getters: {

// ...

doneTodosCount: (state, getters) => {

return getters.doneTodos.length

}

}

## Гетерлер сонымен қатар 2 -ші аргумент ретінде басқа гетерлерді алады:

## Method-Style Access

getters: {

   Creates: (state) => state.posts.filter((todo =>todo/2>= 3),

// Creates: (state) => (id) => state.posts.filter((todo)=>todo/2 >= id),

  },

console.log(this.$store.getters.Creates);

//console.log(this.$store.getters.Creates(10));

// console.log(this.$store.state.post.posts.filter((todo) => todo / 2 >= 3));

## The mapGetters Helper

getters: {

    COUNT: (state) => state.numberState,

    createPostes(state) {

      return state.posts;

    },

  },

import { mapGetters } from "vuex";

 computed: {

    ...mapGetters(["COUNT", "createPostes"]),

  },

    console.log(this.COUNT);

----------------------------------

**#Мутации**

state: {

count: 1

},

mutations: {

increment(state) {

// изменяем состояние

state.count++;

}

}

## store.commit('increment');

## Commit with Payload

mutations: {

increment (state, payload) {

state.count += payload

}

}

store.commit('increment', 10)

## Object-Style Commit

mutations: {

increment (state, payload) {

state.count += payload.amount

}

}

store.commit({

type: 'increment',

amount: 10

})

## Mutations Follow Vue's Reactivity Rules ???

state.obj = { ...state.obj, newProp: 123 }

state-дегі obj-деп аталатын обьектке newProp: 123 -деген элемент қосып жатыр

  state: {

    posts: [1, 4, 34, 7, 3, 6],

  },

  mutations: {

    [SOME\_MUTATION](state) {

      state.posts = [...state.posts, ...[54, 23, 678]];

    },

  },

Данныйға данный жалғау, егер ...- мұны қоймаса обьект немесе массив ішіндегілерді бір-бірден ала алмайды (...array) сяқты функция

state-дегі post-деп аталатын моссифке [54, 23, 678] -деген элемент қосып жатыр

## Using Constants for Mutation Types ??? необизательно

export const SOME\_MUTATION = 'SOME\_MUTATION'

import { SOME\_MUTATION } from './mutation-types'

state: { ... },

mutations: {

// we can use the ES2015 computed property name feature

// to use a constant as the function name

[SOME\_MUTATION] (state) {

// mutate state

}

}

export const SOME\_MUTATION = "SOME\_MUTATION";

import { SOME\_MUTATION } from "../mutation\_types";

 mutations: {

    [SOME\_MUTATION](state, poyload) {

      state.numberState = poyload - 1;

    },

  },

      this.$store.commit("SOME\_MUTATION", 19);

## Mutations Must Be Synchronous ???

mutations: {

someMutation (state) {

api.callAsyncMethod(() => {

state.count++

})

}

}

mutations синхронный болуы шарт егер асинхронный болу керек болса оны action-ға жазамыз

## Committing Mutations in Components

    <button @click="$store.commit('MUTATION\_INCREMENT', 43)"> click </button>

 mutations: {

    MUTATION\_INCREMENT(state, value\_Mutation) {

      console.log(value\_Mutation, state.numberState);

    },

  },

import { mapMutations } from 'vuex'

export default {

methods: {

...mapMutations(['increment', 'incrementBy']), 2-еуінде шақыруға болады

...mapMutations(['', 'incrementBy']), - осылай дұрыс

...mapMutations({add: 'increment'}) Mutations-дағы incrementBy функцияны // add-ға теңестіру

},created() {

// this. increment (1000);

// this.incrementBy(2000);

this.add(2000);

// this.$store.dispatch("increment ", 1000);

},

}

mutations: {

    createPost(state, newPost) {

      state.posts.unshift(newPost);

    },

  },

<form @submit.prevent="submit">

    <button type="submit">Create Post</button>

  </form>

 methods: {

    ...mapMutations(["createPost"]),

    submit() {

      this.createPost({

        title: this.title,

        body: this.body,

      });

        this.title = this.body = ""

    },

  },

On to Actions

store.commit('increment')

actions: {

    increment({ commit }, value) {

      commit("MUTATION\_INCREMENT", value);

    },

  },

---------------------------------------

**#Actions**

state: {

count: 0

},

mutations: {

increment (state) {

state.count++

}

},

actions: {

increment (context) {

context.commit('increment')

}

}

actions: {

increment ({ commit }) {

commit('increment')

}

}

## [#](https://vuex.vuejs.org/guide/actions.html#dispatching-actions)Dispatching Actions

Неге сразу mutation-ға запрос жібере салмаймыз

өйткені, mutation асинхронный функция орындай алмаймыз

ал action -да асинхронный функция болады

store.dispatch('increment')

actions: {

incrementAsync ({ commit }) {

setTimeout(() => {

commit('increment')

}, 1000)

}

}

**object-style dispatch:**

ал action -да данныйды күрделі түрде object түрде жіберуге болады

// dispatch with a payload

store.dispatch('incrementAsync', {

amount: 10

})

 add() {

      this.$store.dispatch("increment3", { amount: 12 });

    },

  increment3({ commit }, value3) {

      commit("MUTATION\_INCREMENT", value3);

      console.log("increment3 - f- " + value3.amount);

    },

// dispatch with an object

store.dispatch({

type: 'incrementAsync',

amount: 10

})

 add() {

      this.$store.dispatch("increment3", {

        type: "MUTATION\_INCREMENT",

        amount: 10,

      });

    },

 actions: {

    increment3({ commit }, value3) {

      commit(value3.type, value3.amount);

      // console.log("increment3 - f- " + value3);

      console.log("increment3 - f- " + value3.amount);

    }

  },

 mutations: {

    MUTATION\_INCREMENT(state, poyload) {

      state.numberState = poyload - 1;

      console.log("poyload - " + poyload);

    },

  },

**Пример**

actions: {

checkout ({ commit, state }, products) { actions -ішінен state шақыру үшін

// save the items currently in the cart

const savedCartItems = [...state.cart.added] ( state.value -дұрысы осы)

// send out checkout request, and optimistically

// clear the cart

commit(types.CHECKOUT\_REQUEST) ??? обьект, обьект ішіндегі элемент арқылы мутация атын берлгілеу

// the shop API accepts a success callback and a failure callback

shop.buyProducts(

products,

// handle success

() => commit(types.CHECKOUT\_SUCCESS), обьект, обьект ішіндегі элемент арқылы мутация атын берлгілеу

// handle failure

() => commit(types.CHECKOUT\_FAILURE, savedCartItems)

)

}

}

## Dispatching Actions in Components

metimport { mapActions } from 'vuex'

hods: {

...mapActions([

'increment', // map `this.increment()` to `this.$store.dispatch('increment')`

// `mapActions` also supports payloads:

'incrementBy' // map `this.incrementBy(amount)` to `this.$store.dispatch('incrementBy', amount)`

]),

...mapActions(['increment','incrementBy' ]), 2-еуінде шақыруға болады

...mapActions(["",'incrementBy' ]), барын ғана шақыруға болады

...mapActions({

add: 'increment' // map `this.add()` to `this.$store.dispatch('increment')`

})

} import {  mapActions } from "vuex";

  methods: {

    ...mapActions(["", "increment3"]),

    add() {

      this.increment3(12);

    },

  },

  methods: {

    ...mapActions({ adde: "increment2" }),

    add() {

      this.increment3(12);

    },

  },

  methods: mapActions(["fetchPosts"]),

async mounted() {

    this.fetchPosts();

  },

## Composing Actions

store.dispatch('actionA').then(() => {

// ...

})

actions: {

actionA ({ commit }) {

return new Promise((resolve, reject) => {

setTimeout(() => {

commit('someMutation')

resolve()

}, 1000)

})

}

}

--------------

actions: {

// ...

actionB ({ dispatch, commit }) {

return dispatch('actionA').then(() => {

commit('someOtherMutation') // Mutation

})

}

}

actions: {

async actionA ({ commit }) {

commit('gotData', await getData())

},

async actionB ({ dispatch, commit }) { dispatch - деген actions бір фуекциясының ішінде тұрып екінші функцияға шықыру, данный жіберу

await dispatch('actionA') // wait for `actionA` to finish

commit('gotOtherData', await getOtherData())

}

}

------------------

actions: {

async loadReactions({ commit }, payload) {

try {

const res = await fetch(

"https://jsonplaceholder.typicode.com/posts?\_limit=5"

);

const posts = await res.json();

commit("SET\_REACTION", posts);

commit("SET\_MESSAGE");

} catch (error) {

console.error(error);

}

},

},

**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

## Module Local State ???

import moduleA from "./moduleA/post";

import moduleB from "./moduleB/post";

Vue.use(Vuex);

export default new Vuex.Store({

  modules: {

    a: moduleA,

    b: moduleB,

  },

});

 methods: {

    add() {

      console.log(this.$store.state.b.numberState);

      console.log(this.$store.state.a.numberState);

    },

  },

[**#**](https://vuex.vuejs.org/guide/modules.html#namespacing) **Namespacing**

  modules: {

    b: moduleB,

    a: moduleA,

  },

export default { // moduleB

  namespaced: true,

  state: {

    numberState: 12823,

  },

  getters: {

    SOUNT: (state) => state.numberState,

  },

};

**1-тәсіл**

    <h1>{{ SOUNT }}</h1>

<!-- <h3>{{ count }}</h3> -->

<h3>{{ numberState }}</h3>

  computed: {

    ...mapGetters("b", ["SOUNT"]),

// ...mapState({

    //   count: (state) => state.b.numberState,

    // }),

    ...mapState("b", ["numberState"]),

  },

**2- тәсіл**

    <h1>{{ cart }}</h1>

  computed: {

    ...mapGetters({ cart: ["b/SOUNT"] }),

  },

  methods: {

    // ...mapActions(["increment", "increment2"]),

    ...mapActions({

      increment: ["b/increment"],

      increment2: ["b/increment2"]

      }),

  ...mapActions('cart',["increment", "increment2"]),

    ...mapMutations("b/subModule", ["MUTATION\_INCREMENTES"]),

  functiona() {

      this.MUTATION\_INCREMENTES(47);

}

}

**modules** ішіндегі **module** - осылай болса төмендегідей

export default {

  namespaced: true,

  modules: {

    subModule: {

      namespaced: true,

      state: {

        subModuleNumber: 9897,

      },

      mutations: {},

      getters: {},

      actions: {},

    },

  },

  state: {

    numberState: 12823,

  },

  mutations: {},

  actions: {},

  getters: {

    SOUNT: (state) => state.numberState,

  },

};

    <h3>{{ subModuleNumber }}</h3>

    <button @click="functiona()">HOME Click</button>

 computed: {

    ...mapState("b/subModule", ["subModuleNumber"]),

  },

  methods: {

    ...mapMutations("b/subModule", ["MUTATION\_INCREMENTES"]),

  functiona() {

      this.MUTATION\_INCREMENTES(47);

}

}

Қалғандарын үйрену  
**Accessing Global Assets in Namespaced Modules**

**{ root : true }** Бір модулдің ішінде тұрып екінші модулдағы функция, данныйларды шақыруғва мүмкіндік береді

Бұлай болған кезде екі модулда бірдей атпен функция жасауға болады екен

      this.$store.dispatch("b/subModule/subModules2", this.number);

export default {// moduleB

  namespaced: true,

  modules: {

    subModule: {

      namespaced: true,

      actions: {

        subModules2({ dispatch }, value) {

          dispatch("a/subModules", value, { root: true });

// root:true - алдынғы путьты қосу, басқа модулдағы функцияны шақыруғa мүмкіндік беру

          console.log("subModules -" + value);

        },

      },

      state: {},

      mutations: {},

      getters: {},

    },

  },

export default { // moduleA

  namespaced: true,

  actions: {

    subModules(\_, value) {

      console.log("subModule +" + value);

    },

}

}

**Register Global Action in Namespaced Modules**

Егер жоғарыдағы мысал бойынша Путь көрсетпейақ басқа модулдағы функцияны шақырғың келсе, сол шақырғың келген функцияның алдына (функцияға) **handler** жалғау керек

      actions: {

        subModules2({ dispatch }, value) {

          dispatch("subModules", value, { root: true });

// root:true - handler қолданғандықтан алдына путь қосу керегі жоқ , басқа модулдағы функцияны шақыруғa мүмкіндік беру

          console.log("subModules -" + value);

        },

      },

 actions: {

    subModules2: {

      root: true,

      handler(\_, value) {

        console.log("subModule +" + value);

      },

    },

}

**Binding Helpers with Namespace**

Модуль ішіндегі модульді шақыратын кезде осы тәсілдерді пайдаланса болады

State - бойынша Шақырудың 3-түрі

    <h3>{{ a }}</h3>

    <h3>{{ b }}</h3>

**1-тәсіл**

 computed: {

    ...mapState({

      a: (state) => state.b.subModule.subModuleNumber,

      b: (state) => state.b.subModule.subModuleNumberb,

    }),

**2-тәсіл**

 computed: {

    ...mapState("b/subModule", {

      a: (state) => state.subModuleNumber,

      b: (state) => state.subModuleNumberb,

    }),

**3-тәсіл**

import { createNamespacedHelpers, mapMutations } from "vuex";

const { mapState } = createNamespacedHelpers("b/subModule");

computed: {

    ...mapState({

      a: (state) => state.subModuleNumber,

      b: (state) => state.subModuleNumberb,

    }),

  },

Getters - бойынша Шақырудың 1-түрі

    <h3>{{ subModuleGetters }}</h3>

    <h3>{{ subModuleGet }}</h3>

**1-тәсіл** болмады

**2-тәсіл**

    ...mapGetters("b/subModule", ["subModuleGetters", "subModuleGet"]),

**3-тәсіл**

import { createNamespacedHelpers} from "vuex";

const { mapGetters } = createNamespacedHelpers("b/subModule");

    ...mapGetters(["subModuleGetters", "subModuleGet"]),

Action - бойынша Шақырудың 3-түрі

**1-тәсіл** болмады

**2-тәсіл**

       ...mapActions("b/modules", ["subModules2", "subModules"]),

**3-тәсіл**

import { createNamespacedHelpers} from "vuex";

const { mapActions} = createNamespacedHelpers("b/subModule");

    ... mapActions (["subModules2", " subModules "]),

**Caveat for Plugin Developers ???`**

## когда вы создаёте [плагин](https://vuex.vuejs.org/ru/guide/plugins.html)  (Plagin продвинутые уревен- де)

Плагин жасаушыға осылай жасасаң Vuex-ке жалғағанда боладтын проблемалар шешіледі

## #Dynamic Module Registration

Вы можете зарегистрировать модуль

// регистрация модуля `myModule`

store.registerModule('myModule', {

// ...

});

// регистрация вложенного модуля `nested/myModule` **???` // модул ішіндегі модул жасау**

store.registerModule(['nested', 'myModule'], {

// ...

});

// ../module/post.js

export default {

  namespaced: true,

  state: {

    dinamicState: 12823,

  },

  mutations: {

    MUTATION\_DINAMIC(state) {

      state.dinamicState = state.dinamicState / 12;

    },

  },

  actions: {},

  getters: {},

};

<script>

import { mapState, mapMutations } from "vuex";

import store from "../store";

import postModule from "../module/post.js";

store.registerModule("postModule", postModule);

export default {

  computed: {

    ...mapState("postModule", ["dinamicState"]),

  },

  methods: {

    ...mapMutations("postModule", ["MUTATION\_DINAMIC"]),

    Dinamic() {

      this.MUTATION\_DINAMIC();

    },

  },

};

</script>

store.unregisterModule(moduleName) Удалить **???`**

store.hasModule(moduleName) арегистрирован ли уже модуль тексеру **???`**

store.registerModule('a', module, { preserveState: true }) - алдыңғы модулді сақтау **???`**

## #Module Reuse

Кейде модульдің бірнеше данасын жасау қажет болуы мүмкін,

runInNewContext в значении false или 'once'

**қай жерде қайталау керек болады білмедім** ???

**бірақ мысал динамик модул бойынша орындалды**

const MyReusableModule = {

state: () => ({

foo: 'bar'

})

// мутации, действия, геттеры...

};

{{ dinamicState }} <br />{{ foo }}<br />

store.registerModule("postModule",postModule.postModule.modules.subModule );

store.registerModule("postModule2", postModule.postModule);

export default {

  computed: {

    ...mapState("postModule", ["dinamicState"]),

    ...mapState("postModule2", ["foo"]),

  },

Vuex - ті басқаша тәсілме қалай қолданасың ???

Vuex - те басқаша тәсілмен export-ты қалай қолданасың ???