**Router** - ді проект создавать етіп жатқанда қосып қою керек

router - CLI да сразу таңдап орнатқан жақсы

router - жеке элемент страницаларына өткенде path: "shop/:id" қою керек "shop/2","shop/5"

сияқты кез-келгенін қабылдай береді

- <router-view>

<router-link :to="**/home**">

**this.$router.push("/home ") - алдына слеш қойсаң роутердегілерді өшіріп басынан жібереді, сен жазғаныңды**

**this.$router.push({name:"One"})**

<router-link :to="/" replace> ??? мағынасы **өзгеру** жазсада, жазбасада болады

router.replace('/') // өзгерту **push орнына қолданса болады**

{ path: '\*' }  либой нәрсе жазсаң осы бетке барады // ошибки 404

{ path: '/user-\*' }  }  // user- осы сөзден кейін либой нәрсе жазсаң осы бетке барады

  { path: "\*",

    name: "Two",

    component: () => import("@/components/Two.vue"),

  },

**params**

<router-link

          to="/one/5"

          :class="{ item\_\_active: $route.params.type === '5' }"

        >

  created() {

    console.log(this.$route.params.id);

  },

 watch: {

    $route(to) {

      console.log(to.params.id);

    },

  },

  props: ['id'],

  created() {

    console.log(this.id);

  },

{

    path: "/one/:type"

  },

<router-link :to="{ name: 'user', params: { id: 123 }}">

router.push({ name: 'user', params: { id: 123 }})

**query**

http://localhost:8080/b/?id=example

    <h2>{{ $route.query.id }}</h2>

{

    path: "/one/?type",

  },

{ path: '/params/:foo/:bar' },

{ path: '/params-with-regex/:id(\\d+)' }, кез келген бүтін сандар болса ғана істейді, теріс сандар немесе нақты сандар (0.5) болса істемейді $route.params.id арқылы алуға болады

{ path: '/asterisk/\*' },

{ path: '/optional-params/:id?' }, - id орнында значение болсада болмасада көрсету

{ path: '/optional-group/(foo/)?bar' } foo/ - бар болсада жоқ болсада көрсету, жақшаға алу, ? міндет, егер ол болмаса /optional-group/bar' осы күйде болсада істей береді

children

{

    path: "/a/:id",

    name: "A",

    component: () => import("@/components/a.vue"),

    children: [

      {

        path: "",

        name: "A",

        component: () => import("@/components/a.vue"),

      },

      {

        path: ":id",

        name: "A",

        component: () => import("@/components/a.vue"),

      },

      {

        path: "a1",

        name: "A1",

        component: () => import("@/components/a1.vue"),

      },

    ],

  },

## router.go(n)

## window.history.go(n). ??? //console арқылы н/е js арқылы беру

router.go(-1) // перейти на одну запись назад, эквивалентно history.back()

router.go(3) // перейти на 3 записи вперёд

router.go(-100 )

router.go(100)

// жазбалар жеткіліксіз болса, ауысу болмайды

Сотметить, что методы навигации Vue router'а (push, replace, go) работают одинаково

во всех режимах (history, hash и abstract)

@click="$router.go(-1)"

# Redirect and Alias

## Redirect

{ path: '/a', redirect: '/b' }

{ path: '/a', redirect: { name: 'foo' }}

{ path: '/a', redirect: to => {

      {

        path: "\*/\*",

        // redirect: '/b'

      },

alias

 {

    path: "/a1",

    name: "A1",

    alias: '/a2', //   а2-ні басқан кезде роутер өзгереді, ішіндегі данный a1 болып тұра береді

    component: () => import("@/components/a1.vue"),

  },

  {

    path: "/a2",

    name: "A2",

    component: () => import("@/components/a2.vue"),

  },

# Passing Props to Route Components

# Passing Props

    <p>{{ propsFunc }}</p>

 props: ["propsFunc"],

  <p>{{ name }}</p>

    <p>{{ id }}</p>

    <p>{{ geens() }}</p>

 props: ["name", "id", "geens"],

 { path: "/a1",

    name: "A1",

    // props: true, // a1-страницадан  props: ["id"], дегенде id-ді ұстай алу үшін

    props: () => ({ propsFunc: "value propsFunc" }), // props арқылы function қолдану

    // props: (route) => ({ propsFunc: route.query.value }), // function арқылы операция жасау

    // alias: '/a2', //   а2-ні басқан кезде роутер өзгереді, ішіндегі данный a1 болып тұра береді

    component: () => import("@/components/a1.vue"),

  },

  { path: "/a2/:id",

    name: "A2",

    props: { name: "kuanyshbek", id: 19982, geens: () => "assets" },// props арқылы object қолдану

    component: () => import("@/components/a2.vue"),

  },

# Route Components ???

- sidebar для боковой панели  ???

main для основного содержимого

автоматически получает имя default.

path: '/'

components: {

default: Foo, // роутер баратын основный компонент

a: Bar, // <router-view name=a/> екіші қосымша терезеден көрсету

b: Baz // <router-view name=b/> үшінші қосымша терезеден көрсету, қосымша терезені бокавой терезеге қоюға болады т.б. болады

}

{  path: "/exam/:id",

    name: "exam",

    components: {

      default: () => import("@/components/exama.vue"),

      a: () => import("@/components/a.vue"),

      b: () => import("@/components/b.vue"),

    },

  },

<router-link to="/exam">Click a2</router-link> <br />

    <router-view />

    <router-view name="a" />

    <router-view name="b" />

во всех режимах (history, hash и abstract) ???

mode: 'history', обыщный (/) - арқылы жазылады http://localhost:8081/exam

mode: 'hash', главный дан кейін (#/) - белгісі қойылады [ http://localhost:8081/#/ exam]

mode: 'abstract', главный дан басқа еш роутер көрінбейді және алынбайды, бірақ кнопка путьтары істейді, обнавить етсе главный роутерге лақтырады

**Мында жазылады:**

const router = new VueRouter({

  mode: "hash",

  base: process.env.BASE\_URL,

  routes,

});

export default router;

meta:{layout:'main'}

  <router-link to="/meta">Click a</router-link> <br />

    <router-link to="/meta2">Click a2</router-link> <br />

    <router-link to="/meta3">Click a3</router-link> <br />

    <components :is="layout"></components>

 <router-view />

import EmptyLayout from "@/layouts/layoutEmpty";

import MainLayout from "@/layouts/layoutMain";

  computed: {

    layout() {

      if (this.$route.meta.layout) {

        console.log(this.$route.meta.layout);

        return this.$route.meta.layout + "-layout";

      } else {

        return "empty-layout";

      }

    },

  },

  components: { MainLayout, EmptyLayout },

  { //router.js

    path: "/meta",

    name: "B",

    meta: { layout: "empty" },

    component: () => import("@/components/a.vue"),

  },

  {

    path: "/meta2",

    name: "B2",

    meta: { layout: "main" },

    component: () => import("@/components/a1.vue"),

  },

  {

    path: "/meta3",

    name: "B3",

    meta: { layout: "main" },

    component: () => import("@/components/a2.vue"),

  },

???

beforeEnter:function

  {

    path: "/resume",

    name: "Resume",

    component: () =>import "../views/resume/resume",

    beforeEnter: isUser,

  },

beforeEnter: hasToken

afterEnter: functionName // роутер странитцасына кірерден бұрын

beforeEnter: functionName // роутер странитцасына кіргенен кейін

beforeLive... : functionName // роутер странитцасынан шығардан бұрын

const isUser = (to, from, next) => {

  let auth = Cookies.get('clagme-type')

  if(auth) next()

  else if(to.path !== '/auth/sign-in') next('/auth/sign-in')

}

to - бармақшы болған, to.path -бармақшы болған страница

from - келгені туралы ақпарат , from.path - қай страницадан келді соны көрсетеді

next('exampleLink') - жібермекші болған страница, қай страницаға лақтырғың келеді, егер

next() - жақша пустой болса келесі страница, яғни бармақшы болған страницаға (to.path) лақтырады

Cookies.get() || Cookies.set() || Cookies.remove()

import Cookies from "js-cookie";

    <button @click="example">Click</button>

    <button @click="deletes">Delete</button>

  example() {

      console.log("example");

      Cookies.set("example", "example-cookes2");

      console.log(Cookies.get("example"));

    },

    deletes() {

      Cookies.remove("example");

    },

router.beforeResolve, router.afterEach

import NProgress from "nprogress"; // библотека

router.beforeResolve((to, from, next) => { // router.afterEach

- роутердің кезкелген страницасына кірген кезде

  if (to.name) {

    NProgress.start();

  }

  next();

});

router.afterEach(() => {// router.afterEach

- роутердің кезкелген страницасында зогруска аяқталып страница ашылған кезде кезде

  setTimeout(() => {

    NProgress.done(); // router зогруска болғанда төбеден көрінетін лента жасаушы библотека

  }, 500);

});

beforeEnter:function - бұл роутерде қолдануы, vuex-те, component-те басқаша қолданады

  {

    path: "/resume",

    name: "Resume",

    component: () =>import "../views/resume/resume",

    beforeEnter: isUser,

  },

beforeEnter: hasToken

afterEnter: functionName // роутер странитцасына кірерден бұрын

beforeEnter: functionName // роутер странитцасына кіргенен кейін

beforeLive... : functionName // роутер странитцасынан шығардан бұрын

const isUser = (to, from, next) => {

  let auth = Cookies.get('clagme-type')

  if(auth) next()

  else if(to.path !== '/auth/sign-in') next('/auth/sign-in')

}

to - бармақшы болған, to.path -бармақшы болған страница

from - келгені туралы ақпарат , from.path - қай страницадан келді соны көрсетеді

next('exampleLink') - жібермекші болған страница, қай страницаға лақтырғың келеді, егер

next() - жақша пустой болса келесі страница, яғни бармақшы болған страницаға (to.path) лақтырады