**懒加载：**

把不同路由对应的组件分割成不同的代码块儿，然后当路由被访问的时候才加载对应的组件。

实现方式:

在大型应用中，我们可能需要将应用拆分为多个小模块，按需从服务器下载。为了进一步简化，Vue.js 允许将组件定义为一个工厂函数，异步地解析组件的定义。Vue.js 只在组件需要渲染时触发工厂函数，并且把结果缓存起来，用于后面的再次渲染

***1.vue异步组件：***

例：在layout.vue中

..............

components:{//异步组件

headerNav:(resolve,reject)=>{

setTimeout(()=> {

resolve(header)

},2000)

}

}

.....................

这里用setTimeout模拟当从服务器拿到数据的时候，渲染组件header。

***2.webpack代码分割功能：***

*2.1* require.ensure代码分割

require.ensure（1.**依赖**，2.**回调函数**，3.**chunk名字**）

1.依赖为数组 3chunk名字相同名字的组件会被打包在同一个文件里

在路由中：

..............

/\*import layout from '@/views/layout'\*/

/\*import doc from '@/views/backend/doc'\*/

/\*import project from '@/views/backend/project'\*/

/\*import workbench from '@/views/backend/workbench'\*/

/\*webpack代码分割功能和vue按需加载\*/

let layout=(resove) => {

return **require.ensure**([],() => {

resove(require('@/views/layout'))

})

}

let doc=(resove) => {

return **require.ensure**([],() => {

resove(require('@/views/backend/doc'))

})

}

let project=(resove) => {

return **require.ensure**([],() => {

resove(require('@/views/backend/project'))

},'abc')

}

let workbench=(resove) => {

return **require.ensure**([],() => {

resove(require('@/views/backend/workbench'))

},'abc')

}

...............

上面project和workbench的第三个参数相同即将两个组件的打包在一个js文件里了

2.2 import函数

可以将上面分割代码的方式简写：

let layout=(resove) => {

return import('@/views/layout')

}

let doc=(resove) => {

return import('@/views/backend/doc')

}

let project=(resove) => {

return import('@/views/backend/project')

}

let workbench=(resove) => {

return import('@/views/backend/workbench')

}

注意：使用import就不能将两个或者多个文件打包在一起了。import目前不支持第二个参数