

keep-alive

缓存组件，提升性能 被缓存的组件的created只会在首次进入页面的时候执行

Actived()钩子会在每次进入都执行

场景： 从首页进入详情页，每次都是同一个ID 就不用发送请求了。

Scoped()

实现原理：给元素节点新增data-私有属性，从而实现CSS层面的样式隔离

样式穿透

SCSS 父元素 /deep/ 子元素

Stylus 父元素 /deep/ 子元素; >>>

Computed

特点： 基于响应式依赖进行缓存 只在属性发生改变时候 才会重新求值

Vuex的持久化存储

1： localstorage

2： 插件

Vue-router的实现原理

1： Vue.router 是 Vue的一个插件 遵守插件的注册方法 内部实现了install方法；

2： 两种模式 hash history 对应两种实现方式

Hash： window.location.hash拿到当前的url

监听：onhashchange事件

特点：Hash不会包含在HTTP请求中 不会重新加载页面,发起请求

所以使用hash模式，即使在地址栏输入地址 后面的参数不会包含在http请求中，后端始终只需要返回一个HTML即可

History：HTML5History pushState() replaceState({stateObj},pagename, document)

特点： 修改浏览器的历史记录栈 URL改变 但是不会刷新页面, 不会发送请求，会加载document

监听事件： onpopState监听history变化 location.pathname 拿到当前变化的url

在vue中 ，history模式下，加入URL栏重新输入了地址会向服务器发起请求，参数包含在http请求中，假若服务端没有这个资源，会返回404，not found；所以服务器需要返回一404页面；

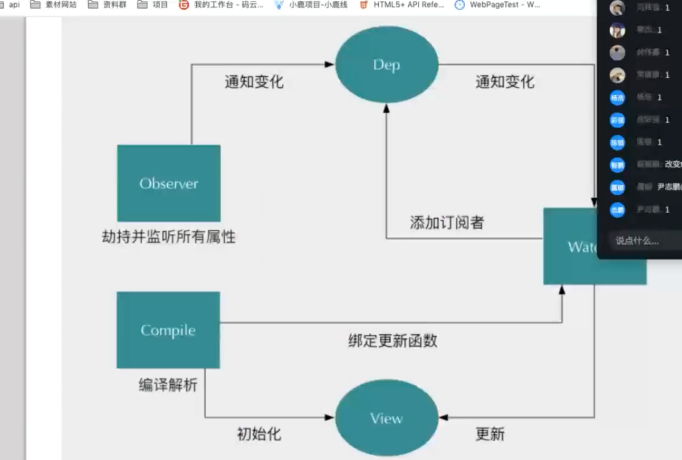
3： 监听“hashchange事件 判断地址栏的改变

4： 注册了两个全局组件 router-view router-link





MVVM



$el 在有值的时候 和$mounted()那个优先级高 ？？

优先级排序 ： el > $mounted(‘dom’)

原因： el是直接在new vue的option里面 ； $mounted（）是在实例化vue后调用的

两者只需要写一个即可

vue会在挂载时，将el /mounted(‘dom’) 传入的DOM的innerHTML全部替换掉

new Vue({

el: "#app1",

render: (h) => h(App),

}).$mount("#app");

Template and Render 同时存在那个优先级高

Render: 的优先级

路由守卫的三种方式

beforeEach() afterEach()

路由独享守卫 beforeEnter() 只有在从一个不同的路由导航时才触发

组件守卫 beforeRouterEnter() 不能访问this ；

在渲染该组件对应的路由被验证前调用

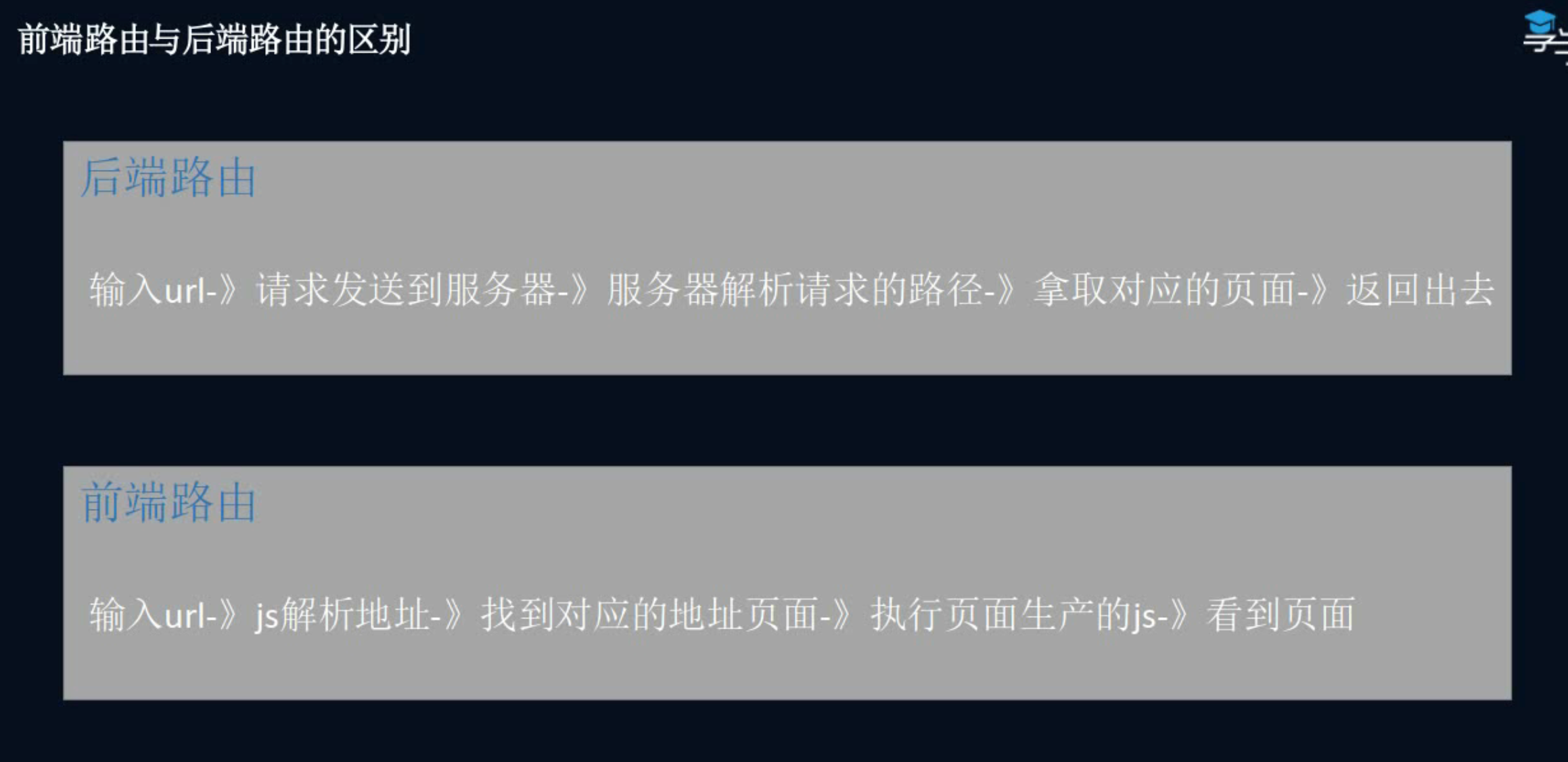
beforeEnterUpdate()

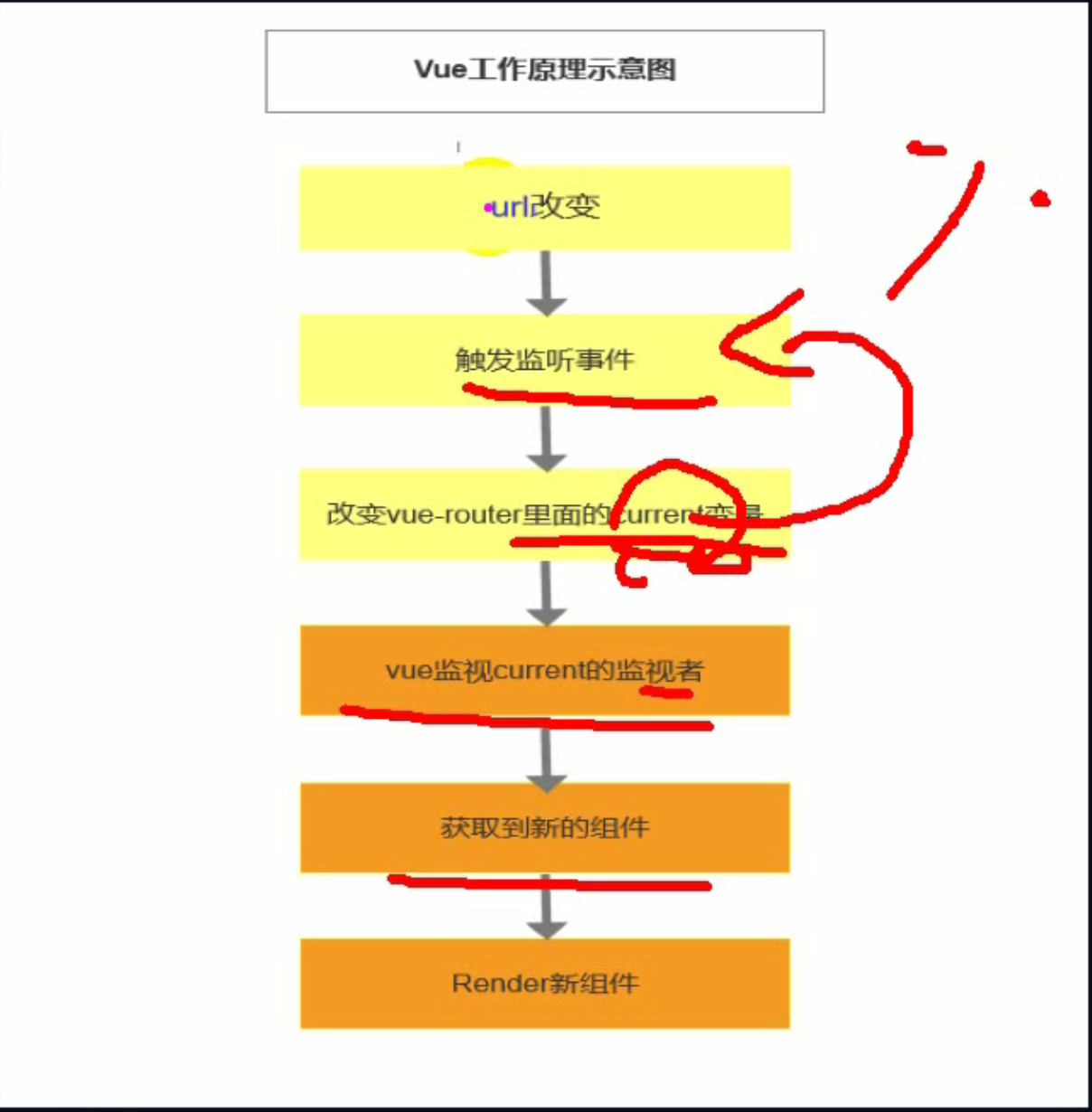
beforeRouteLeave()

$route的实现方式

$store 的实现方式

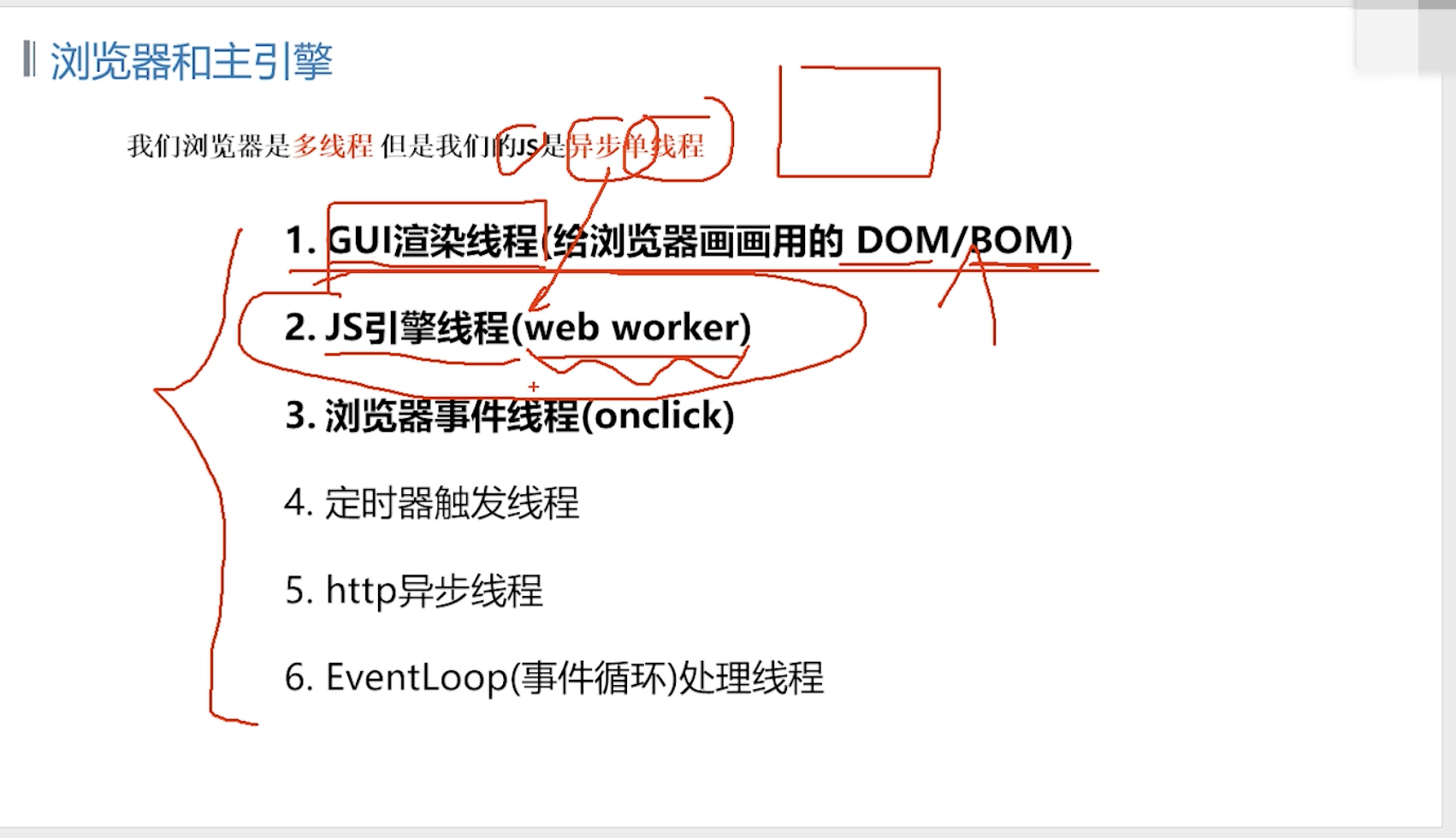
前后端路由的原理



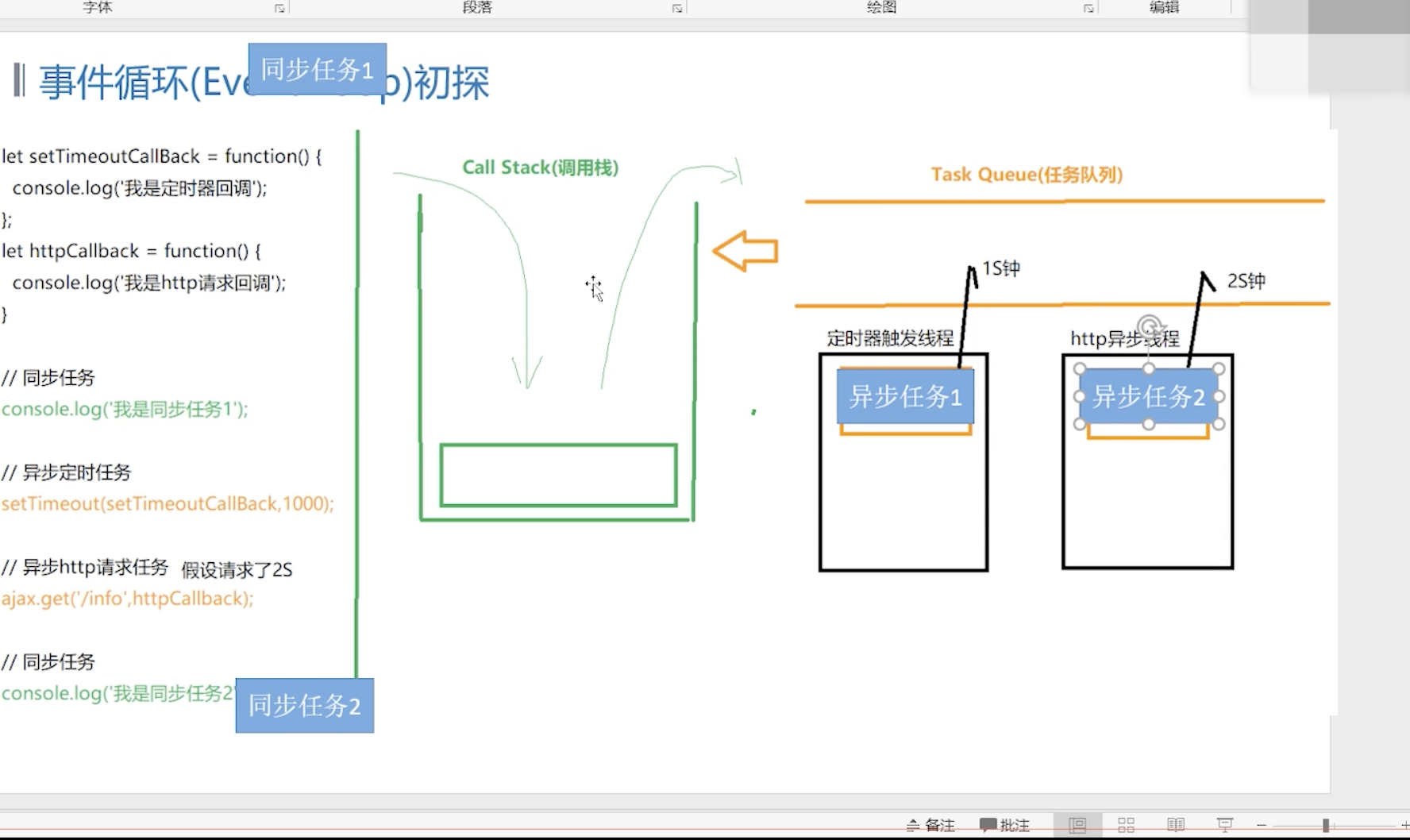


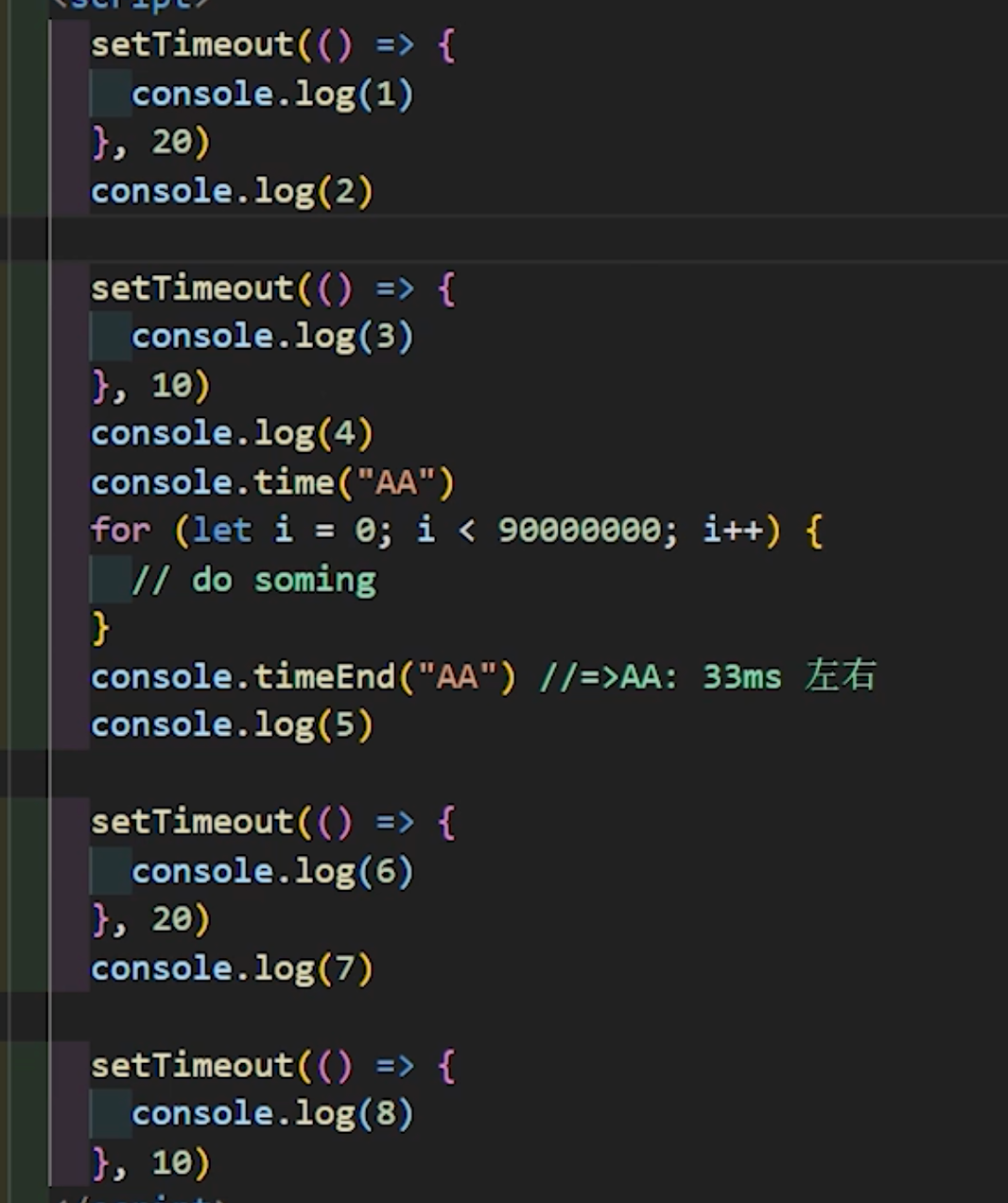
为啥要了解DOM diff 对于实际的开发有啥帮助吗？ 有啥可以借鉴的思想吗？可以怎样将这些思想运用在开发中

diff算法，精细化对比；实现最小量更新

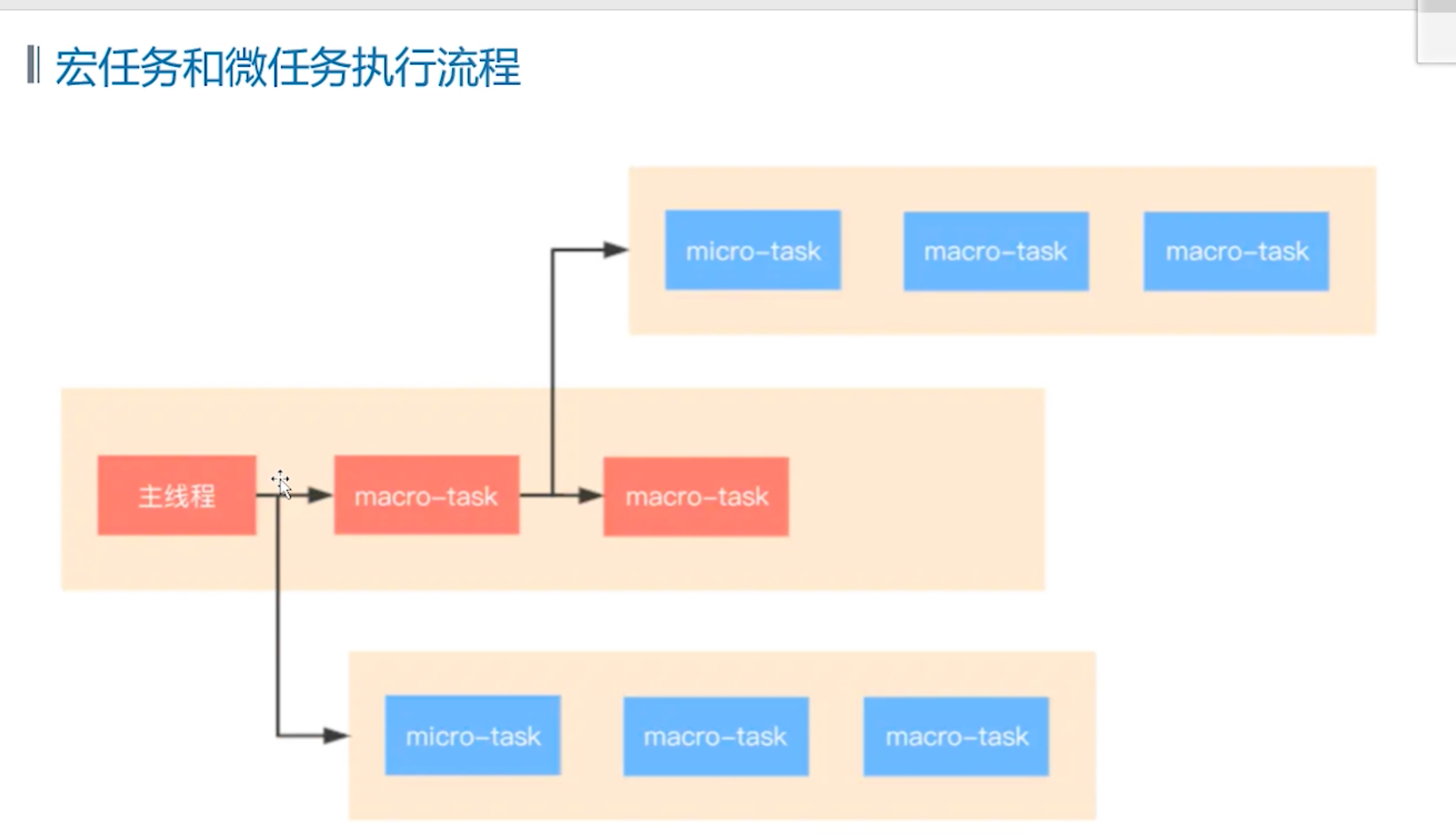


常驻线程 1，2，3





1： 如果主线程的代码运行收到阻塞，假设阻塞了33秒，那本该在10秒，15秒， 20秒后就执行的异步代码会在执行栈清空后，同时立即执行



宏任务和微任务的嵌套

微任务和微任务的嵌套