Vue考点

指令，插值

插值 {{}}

动态属性 v-bind:id

v-html直接插入原始html，有XSS风险，覆盖掉子元素

computed和watch

* computed有缓存，data不变则不会重新计算
* watch拿不到引用类型的oldVal，因为oldVal和Val指向的是同一个内存地址
* watch可以开启深度监听(deep:true)

使用动态属性，驼峰式写法

动态class属性可以是一个对象，也可以是一个数组， :style里面传入的对象必须使用驼峰式写法。

v-if和v-show的区别是v-if会添加和删除dom元素，更消耗性能。而v-show只是改变display:none

**循环列表渲染**

v-for遍历对象。 V-for和v-if不能一起使用的

数组是(item,index)

对象是(val,key,index)

Event参数和自定义参数

* 可以$event传event参数过去
* Vue的$event参数是原生的event参数，可以通过event.\_\_proto\_\_.constructor)来判断，事件被挂载到当前元素。

事件修饰符，按键修饰符

* .prevent 就是preventDefault()
* .stop就是stopPropagation.
* .capture就是addEventlistener里面的第三个参数设置为true。采用事件捕获
* .self就是当event.target等于自身时才触发
* 按键修饰符包括一些.ctrl（包括alt或shift被按下都触发）. ctrl.exact
* v-model lazy防抖
* v-model.trim直接对字符串做trim处理
* v-model.number转化成数据
* textArea不能用插值

Vue父子组件通讯

Props和$emit: v-on或者@事件名来监听一个函数。 然后子组件通过this.$emit(函数名字，参数)触发父组件监听的函数。

自定义事件

兄弟组件之间通过自定义事件进行通讯。

一个组件通过event.$on(事件名，监听函数)来绑定自定义事件。

然后另外一个组件通过event.$emit来触发自定义事件。

这里的event是来自于同一个vue实例。

可以通过event.$off()来解绑，否则造成内存泄露。

组件生命周期

自定义v-model

基本思路就是子组件通过this.emit触发input事件，来完成对父组件v-model的渲染。

一个组件上的v-model会默认使用value的prop和名为input的事件

但是像input

Model选项是用来避免冲突的。

$nextTick

Vue是异步渲染的框架，data改变之后，DOM不会立刻渲染。 $nextTick会在DOM渲染之后会触发，以获取最新的DOM节点。

Data改变之后，拿不到改变之后渲染的DOM的元素，可以使用this.$nextTick( ()=>{

}

)来获取下一次渲染之后的DOM元素。

页面渲染时会将data修改做整合，多次data修改只会渲染一次。

Slot

普通slot的使用方法

Slot标签会显示上级组件传入Slot的值。也可以设置默认内容（当父级组件没传入值的时候显示）

作用域插槽

动态，异步组件

Keep-alive

mixin

vue动态组件

根据:is 来包裹x

**异步组件**

Import()函数

按需加载 () => {

Return import(‘’)

}

异步加载就是什么时候用，什么时候加载，再接收该组件的js文件

**Keep-alive 缓存组件**

看似是动态组件，但是好处是频繁切换。不需要重复渲染。

动态组件，一个组件会触发频繁的创建和销毁。

v-show是css的控制

tab切换

用<keep-alive>包裹组件。是在框架层面进行JS对象的渲染。

Mixin的用处就是抽离相同逻辑

可以通过mixins属性添加，可以是个数组

Mixins实际上返回的是一个vue的实例格式

缺点：

变量来源不明确，不利于阅读

多mixin会造成命名冲突

Mixin和组件可能出现多对多的关系，复杂度较高

数据渲染视图

Vue MVVM

React SetState()

VM是个连接层。