一. Vue组件

1. axios与fetch实现数据请求

(1)fetch

why:

XMLHttpRequest 是一个设计粗糙的 API，配置和调用方式非常混乱，而且基于事件的异步模型写起来不友好。

兼容性不好

polyfill:

https://github.com/camsong/fetch-ie8

```
//get
fetch("**").then(res=>res.json()).then(res=>{console.log(res)})
fetch("**").then(res=>res.text()).then(res=>{console.log(res)})
//post
fetch("**",{
method:'post',
headers: {
"Content-Type": "application/x-www-form-urlencoded"
},
 body: "name=kerwin&age=100"
 }).then(res=>res.json()).then(res=>{console.log(res)});
 fetch("/users",{

 method:'post',
 // credentials: 'include',
 headers: {
 "Content-Type": "application/json"
 },
 body: JSON.stringify({
 name:"kerwin",
 age:100
 })
 }).then(res=>res.json()).then(res=>{console.log(res)});
```

注意：

Fetch 请求默认是不带 cookie 的，需要设置 fetch(url, {credentials: 'include'})

(2) axios

```
axios.get("") promise对象
axios.post("") promise对象
axios.put("")
axios.delete("")

axios({
url:"/gateway?type=2&k=3553574",
headers:{
'X-Client-Info': '{"a":"3000","ch":"1002","v":"1.0.0","e":"1"}',
 'X-Host': 'mall.cfg.common-banner'
 }
 }).then(res=>{
 console.log(res.data);
 })

 返回的数据会被包装

 {
 *:*
 data:真实后端数据
 }
```

2. 计算属性

复杂逻辑,模板难以维护

(1) 基础例子

(2) 计算缓存 VS methods

-计算属性是基于它们的依赖进行缓存的。

-计算属性只有在它的相关依赖发生改变时才会重新求值

(3) 计算属性 VS watch

- v-model

3. Mixins

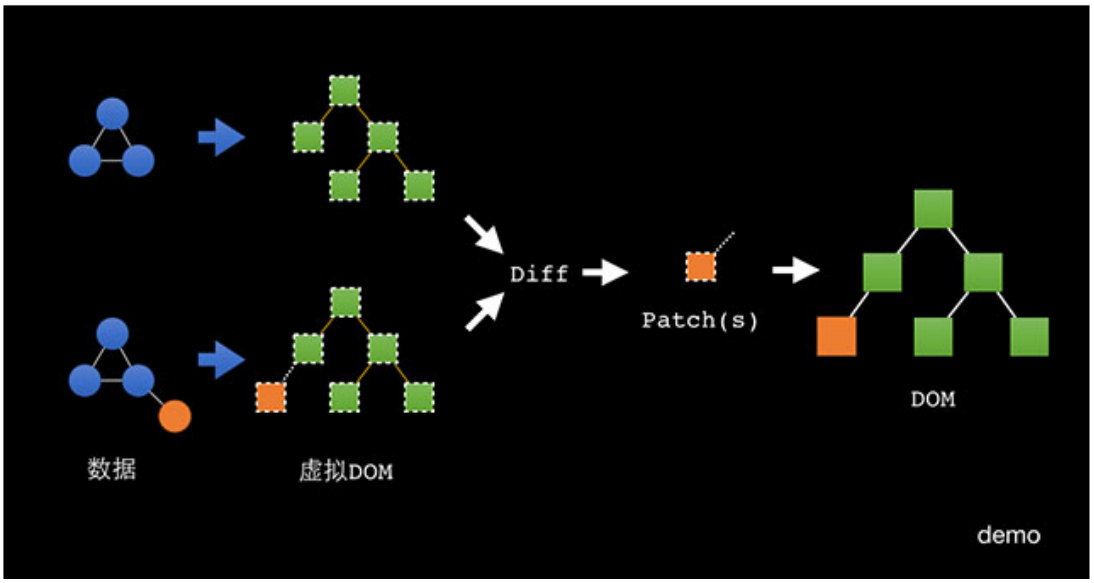
混入 (mixins) 是一种分发 Vue 组件中可复用功能的非常灵活的方式。

混入对象可以包含任意组件选项。

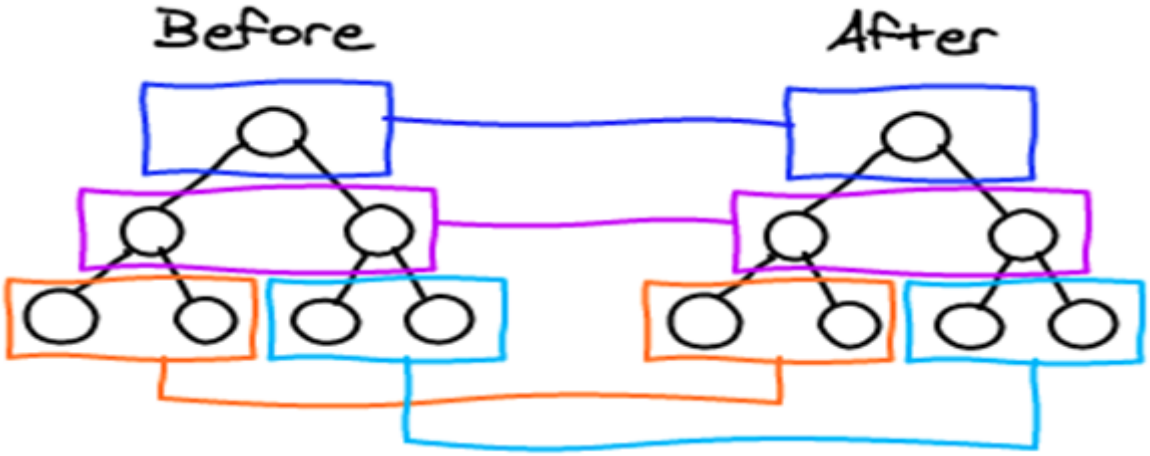
当组件使用混入对象时，所有混入对象的选项将被混入该组件本身的选项。

> https://cn.vuejs.org/v2/guide/mixins.html#%E5%9F%BA%E7%A1%80

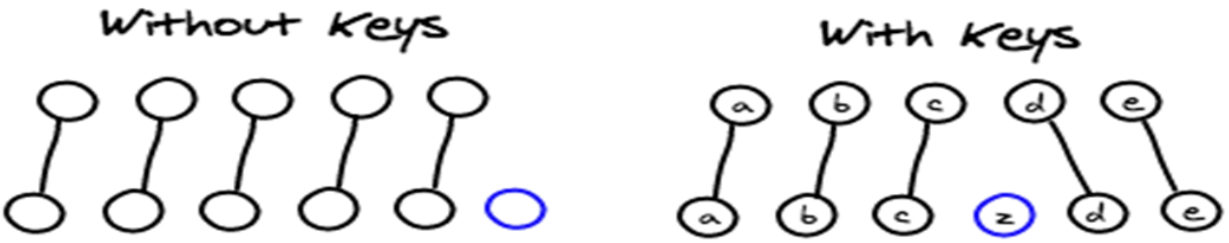
4. 虚拟dom与diff算法 key的作用



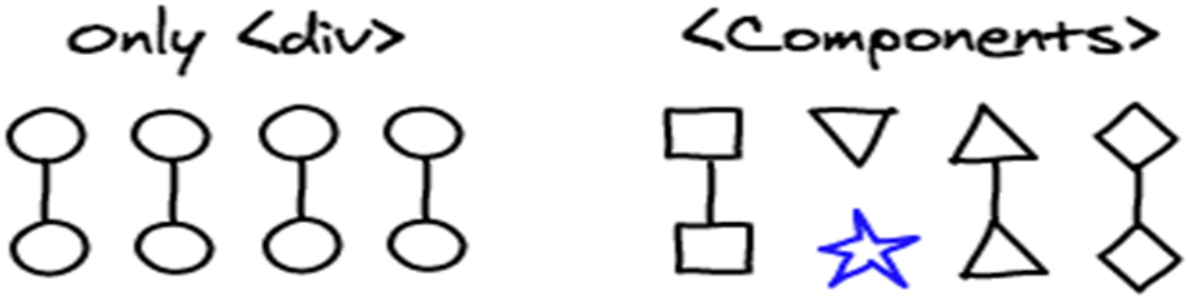
（1）把树按照层级分解



(2) 同key值对比



(3) 同组件对比



5. 组件化开发基础

扩展 HTML 元素，封装可重用的代码

6. 组件注册方式

a.全局组件

Vue.component

b.局部组件

7. 组件编写方式与Vue实例的区别

*自定义组件需要有一个root element

*父子组件的data是无法共享

*组件可以有data，methods,computed....,但是data 必须是一个函数

8. 组件通信

i. 父子组件传值 (props down, events up)

ii. 属性验证

props:{name:Number}

Number,String,Boolean,Array,Object,Function,null(不限制类型)

iii. 事件机制

a.使用 $on(eventName) 监听事件

b.使用 $emit(eventName) 触发事件

iv. Ref

<input ref="mytext"/> this.$refs.mytext

v. 事件总线

var bus = new Vue();

* mounted生命周期中进行监听

9. 动态组件

*<component> 元素，动态地绑定多个组件到它的 is 属性

*<keep-alive> 保留状态，避免重新渲染