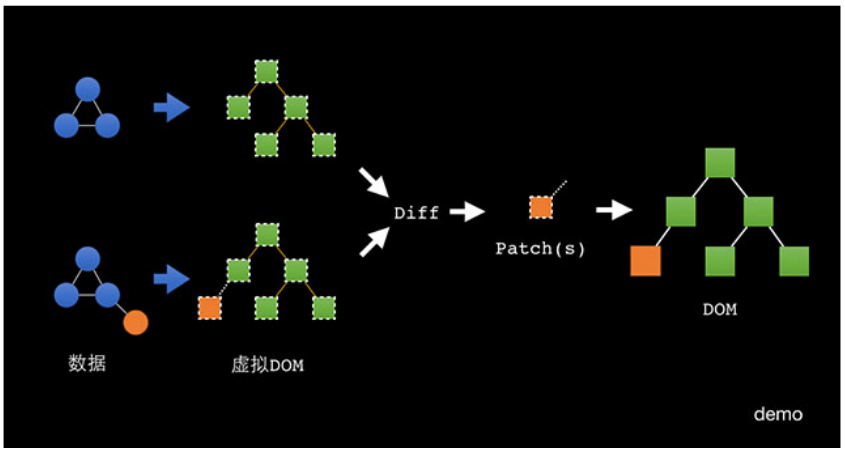
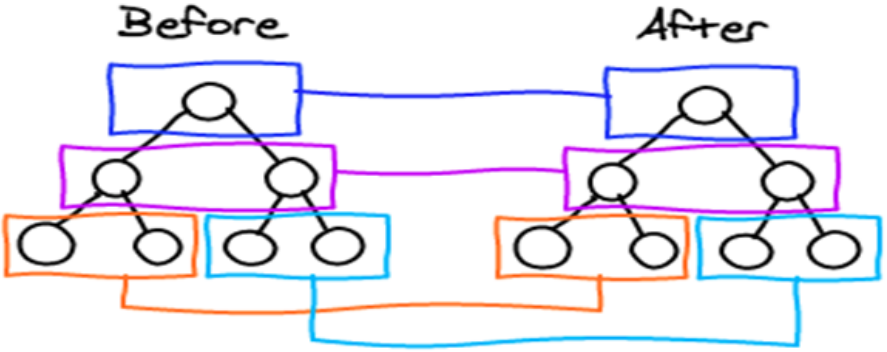
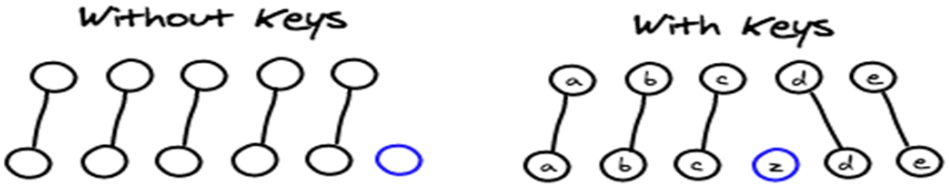
1. 虚拟dom与diff算法 key的作用



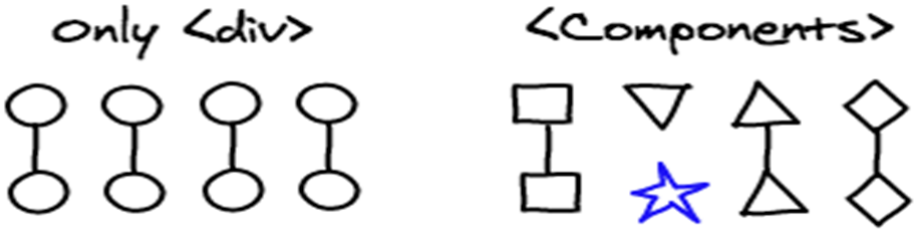
（1）把树按照层级分解



(2) 同key值对比



(3) 同组件对比



2. 为什么组件化

扩展 HTML 元素，封装可重用的代码

3. 组件注册方式

a.全局组件

全局注册

到目前为止，我们只用过 `Vue.component` 来创建组件：

```
Vue.component('my-component-name', {
  // ... 选项 ...
})
```

b.局部组件

局部注册

```
var ComponentA = { /* ... */ }
var ComponentB = { /* ... */ }
var ComponentC = { /* ... */ }
```

然后在 `components` 选项中定义你想要使用的组件：

```
new Vue({
  el: '#app',
  components: {
    'component-a': ComponentA,
    'component-b': ComponentB
  }
})
```

4. 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

5. 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

vi. 问题汇总：

（1）*属性可不可以修改?

严格来说,Vue子组件不能随便更改父组件传递过来的属性,但是可以通过
props 配合 $emit 改变父组件传入的值
//父组件
<my-input :warning.sync="warning" />
在父组件传入值时，需要有xxx.sync修饰符；

//子组件
$emit('update:warning',val)
子组件可以在$emit('update:xxxx',val)，派发事件修改传入的值；

（2）*v-once 用在组件上有什么用?

渲染普通的 HTML 元素在 Vue 中是非常快速的，但有的时候你可能有一个组件，这个组件包含了**大量**静态内容。在这种情况下，你可以在根元素上添加 `v-once` attribute 以确保这些内容只计算一次然后缓存起来，就像这样：

```
Vue.component('terms-of-service', {
  template: `
    <div v-once>
      <h1>Terms of Service</h1>
      ... a lot of static content ...
    </div>
  `
})
```

模板不更新了

6. 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

7. slot插槽 (内容分发)

*混合父组件的内容与子组件自己的模板-->内容分发

*父组件模板的内容在父组件作用域内编译；子组件模板的内容在子组件作用域内编译。

a. 单个slot

b. 具名slot

对于一个带有如下模板的 `<base-layout>` 组件：

```
<div class="container">
  <header>
    <slot name="header"></slot>
  </header>
  <main>
    <slot></slot>
  </main>
  <footer>
    <slot name="footer"></slot>
  </footer>
</div>
```

在向具名插槽提供内容的时候，我们可以在一个 `<template>` 元素上使用 `v-slot` 指令，并以 `v-slot` 的参数的形式提供其名称：

```
<base-layout>
  <template v-slot:header>
    <h1>Here might be a page title</h1>
  </template>

  <p>A paragraph for the main content.</p>
  <p>And another one.</p>

  <template v-slot:footer>
    <p>Here's some contact info</p>
  </template>
</base-layout>
```

注意 v-slot 只能添加在 <template> 上，文本节点也可以当具名插槽内容插入

8. transition过渡

Vue 在插入、更新或者移除 DOM 时，提供多种不同方式的应用过渡效果。

(1)单元素/组件过渡

* css过渡

* css动画

* 结合animate.css动画库

```
<transition
  name="custom-classes-transition"
  enter-active-class="animated tada"
  leave-active-class="animated bounceOutRight"
>
  <p v-if="show">hello</p>
</transition>
```

(2) 多个元素过渡(设置key)

*当有相同标签名的元素切换时，需要通过 key 特性设置唯一的值来标记以让 Vue 区分它们，否则 Vue 为了效率只会替换相同标签内部的内容。

mode:in-out ; out-in

(3)多个组件过渡

(4)列表过渡(设置key)

*<transition-group>不同于 transition， 它会以一个真实元素呈现：默认为一个 。你也可以通过 tag 特性更换为其他元素。

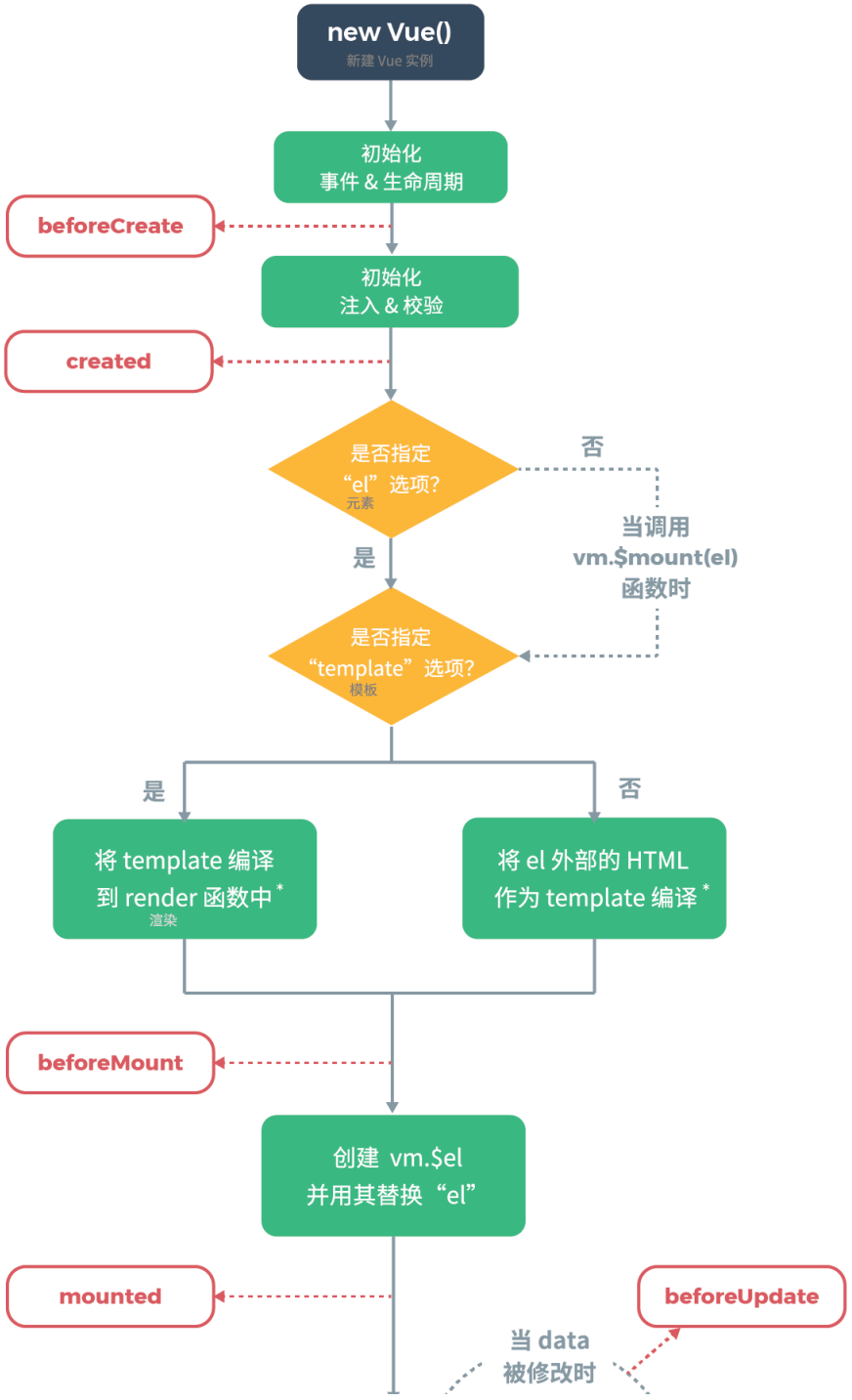
* 提供唯一的 key 属性值

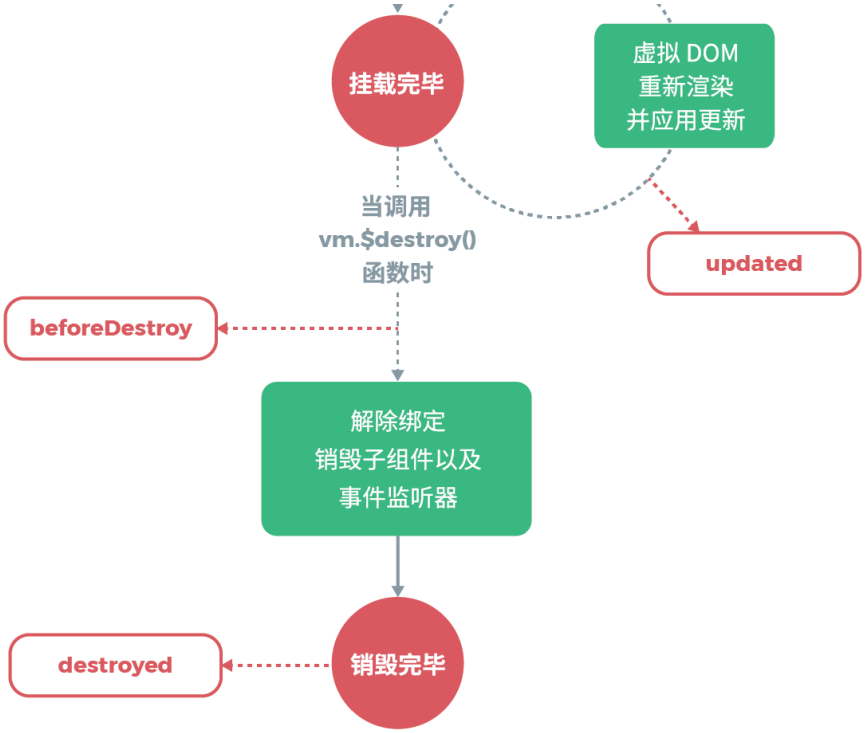
9. 生命周期

i. 生命周期各个阶段

> https://cn.vuejs.org/v2/guide/instance.html#%E7%94%9F%E5%91%BD%E5%91%A8%E6%9C%9F%E5%9B%BE%E7%A4%BA

ii. 生命周期钩子函数的触发条件与作用





* 如果使用构造生成文件（例如构造单文件组件），
模板编译将提前执行

10. swiper学习

> https://www.swiper.com.cn/

11. 自定义组件的封装

自定义封装swiper组件（基于swiper）

注意： 防止swipe初始化过早