vue基础 - day02

学习目标

Learning Objectives

1. 掌握基础指令，完成列表渲染和样式处理
2. 掌握计算属性，实现对数据的逻辑处理
3. 掌握侦听器使用，实现对数据改变的监听

目录

Contents

案例效果：

编号	科目	成绩	考试时间	操作
1	语文	39	2022-06-05 06:12:11	删除
2	数学	100	2022-06-05 06:44:51	删除
总分：139　　平均分：69.50				

科目：请输入科目

分数：请输入分数

添加

目录

Contents

Vue指令-v-for

作用：可以遍历 数组 或者 对象，用于渲染结构

遍历数组语法：

- v-for="item　in　数组名"
- v-for="(item，index)　in 数组名"

遍历对象语法：

- v-for = "(value，key)　in　对象名"

遍历数字

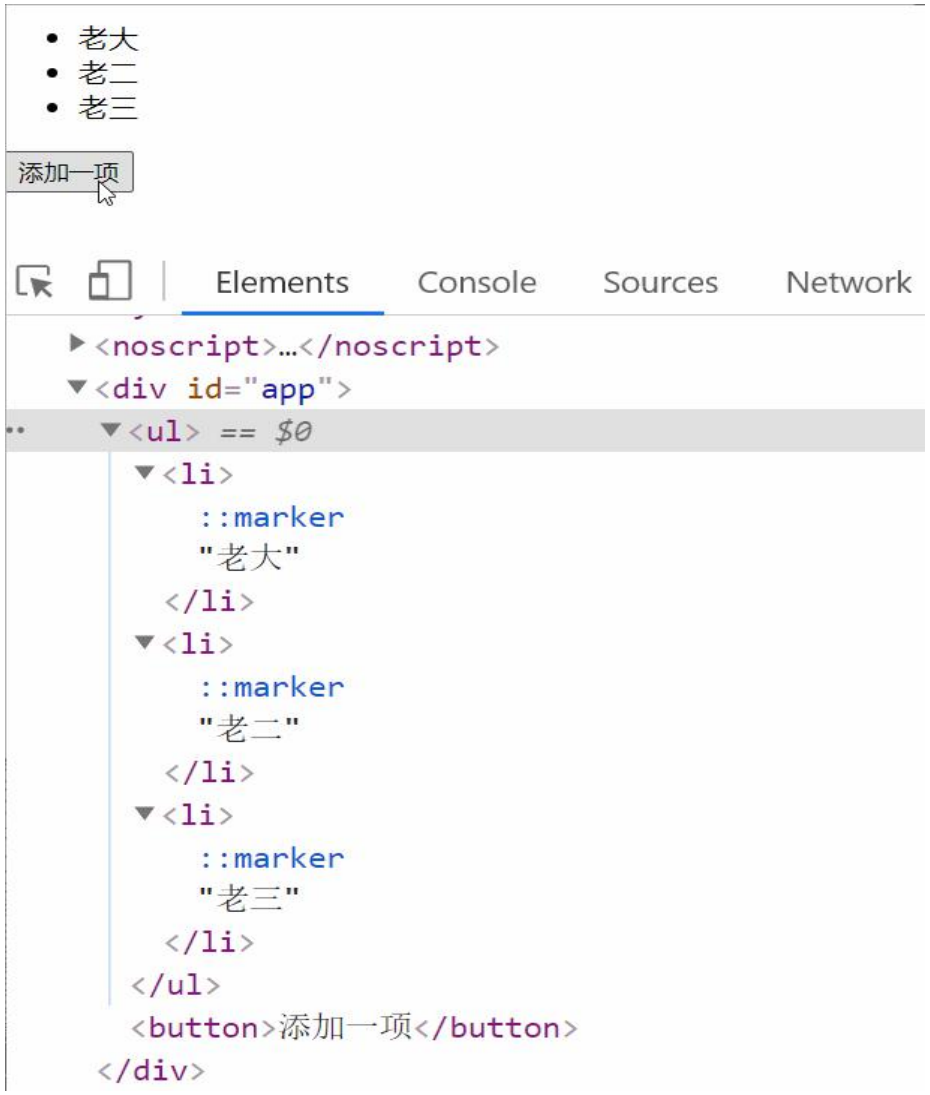
- v-for = "item in 数字"

问题1: v-for数组变化（增加一项，删除一项），会更新页面吗?

问题2: 那么数组改变后，是如何更新的呢?

vue的就地复用策略

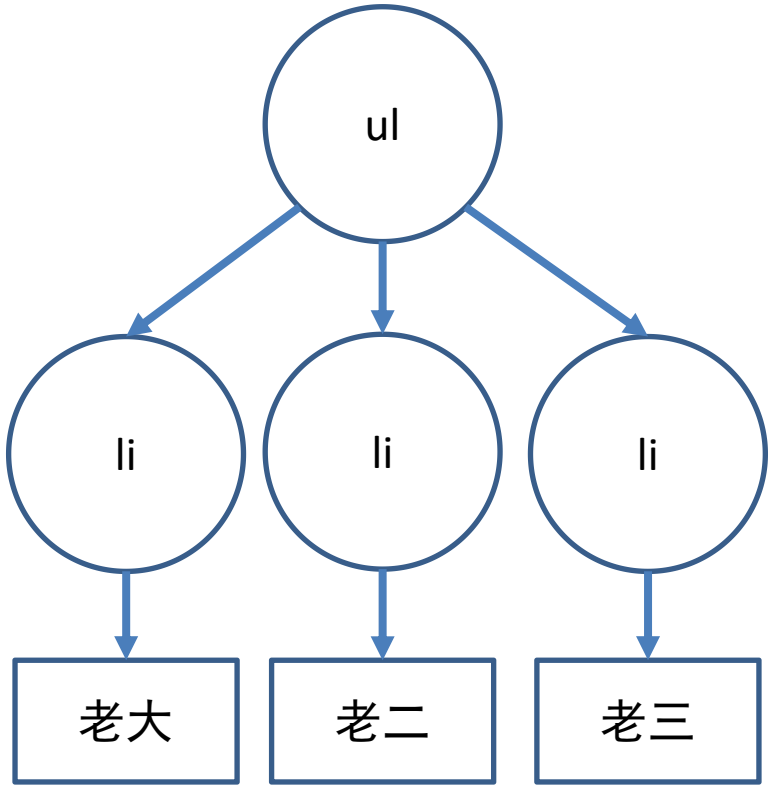
思考：当往数组中加一项，是如何更新的。



vue的就地复用策略

思考：当往数组中加一项后是如何更新的。

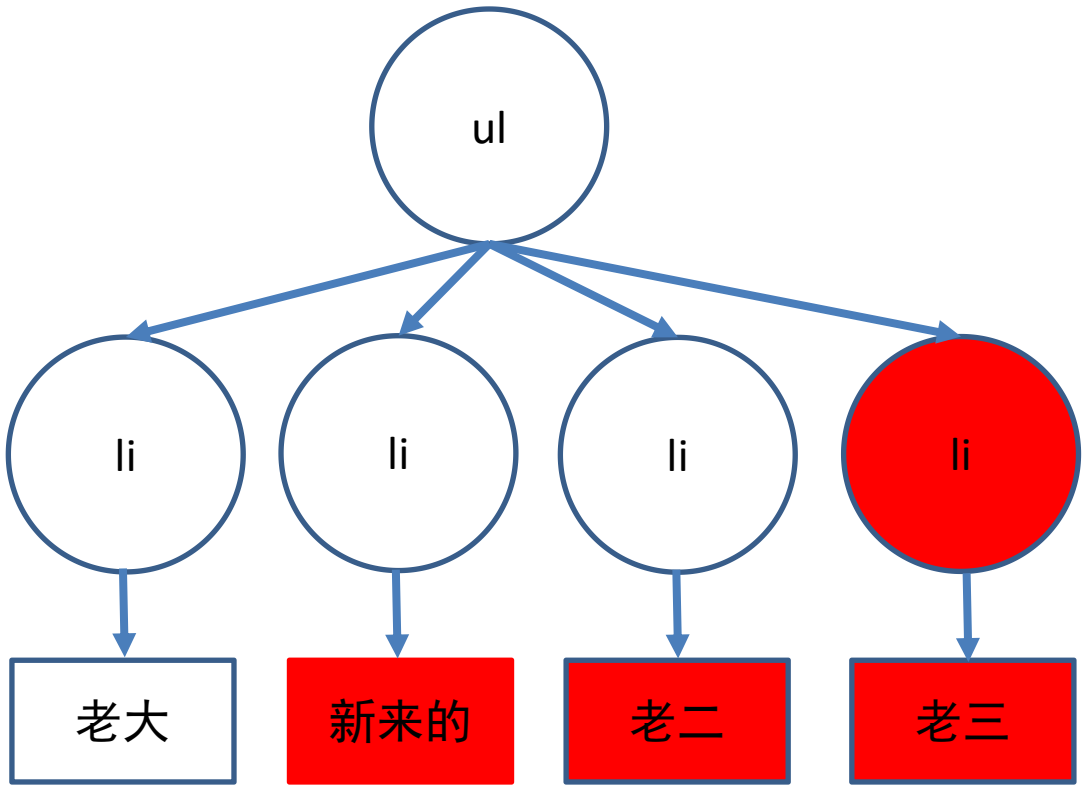
旧-虚拟DOM结构




```
arr: ["老大", "老二", "老三"]
```

就地复用：Vue会尽可能的就地（同层级，同位置）
对比虚拟dom，复用旧dom结构，进行差异化更新。

新-虚拟DOM结构




```
arr: ["老大", "新来的", "老二", "老三"]
```

小结

1. 什么是 vue 的就地复用策略呢？

Vue会尽可能的就地（同层级，同位置），

对比虚拟dom，复用旧dom结构，进行差异化更新。

2. 就地复用的好处是什么？

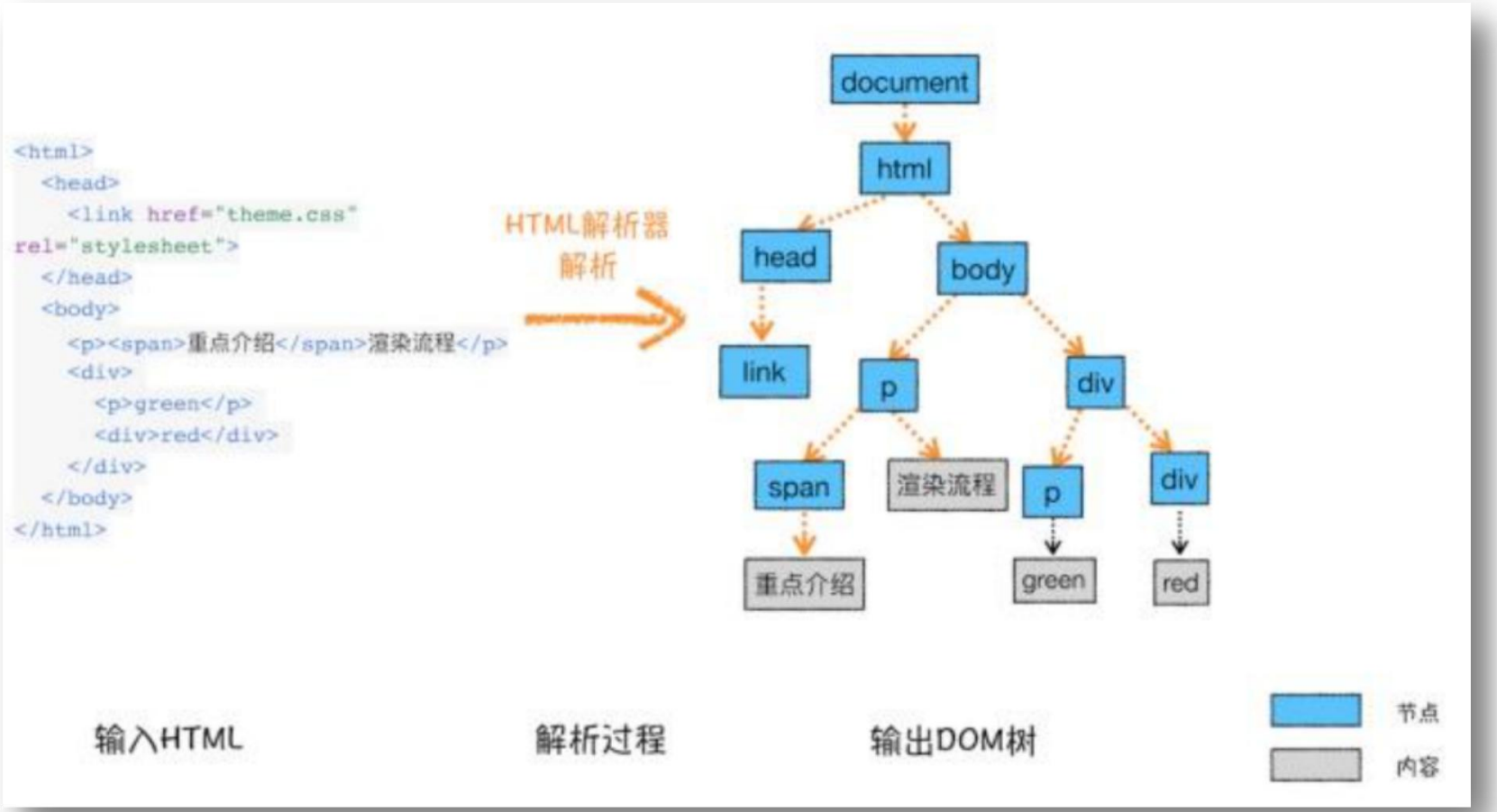
可以复用旧的dom结构，更新高效！

什么是虚拟dom呢？

对比虚拟dom？差异化更新？ 为啥不对比现成的真实 dom 呢？

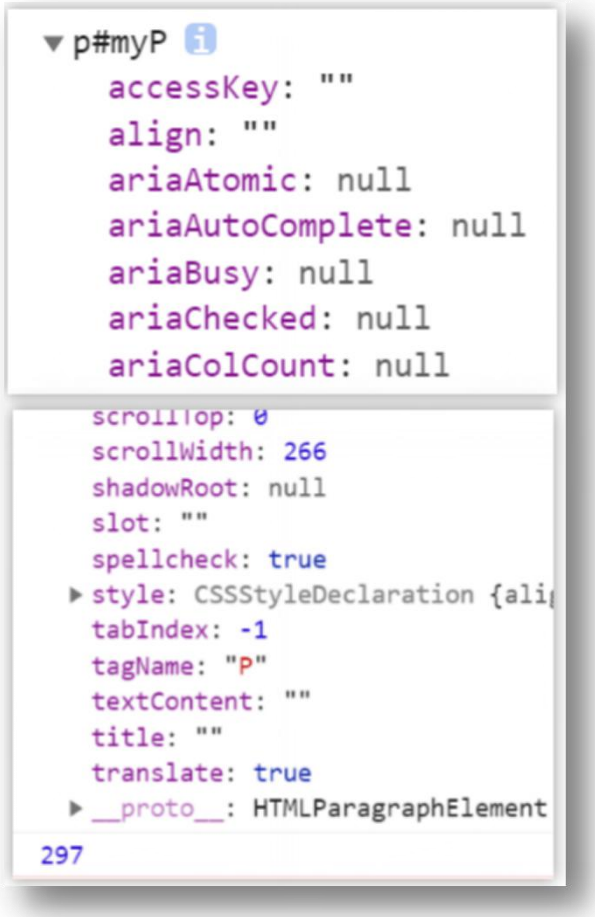
虚拟dom - 页面dom结构是一个树形结构

html 渲染出来的 真实dom树，是个树形结构（复杂）。每个标签，都只是树的某个节点。



虚拟dom - 每个小真实dom节点很复杂

每个标签，虽然只是树形结构的一个小节点，但属性也非常多。=> 遍历真实dom找差异，非常费时！



真实DOM属性过多，有很多无用的属性，无需遍历对比

如何优化呢？对比属性少的虚拟dom！

虚拟dom - 是什么

虚拟dom：本质就是 保存节点信息, 描述真实dom的 JS 对象

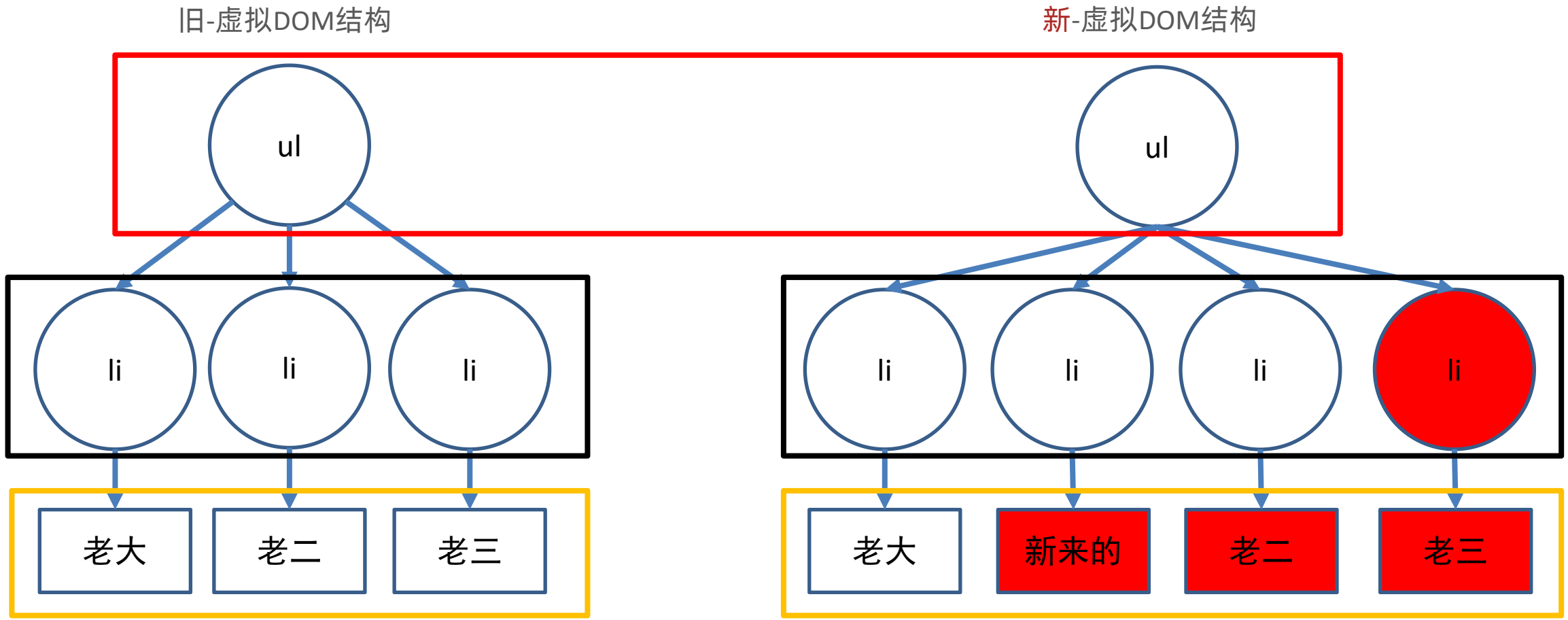
```
<template>
    <div id="box">
        <p class="my_p">123</p>
    </div>
</template>
```

```
const dom = {
    type: 'div',
    attributes: [{id: 'box'}],
    children:[{
        type: 'p',
        attributes: [{class: 'my_p'}],
        text: '123'
    }]
}
```

虚拟dom（一个个js对象）：

可以用最少的属性结构，描述真实的dom

虚拟dom - 对比更新



内存中创建虚拟dom，快速比较变化，给真实DOM打补丁(更新)

具体的对比算法，下一节讲解

小结

1: 对比虚拟dom？ 为啥不对比 真实 dom 呢？

真实dom太复杂，对比性能效率低

2: 什么是虚拟dom呢？

虚拟dom就是描述真实dom的 js 对象

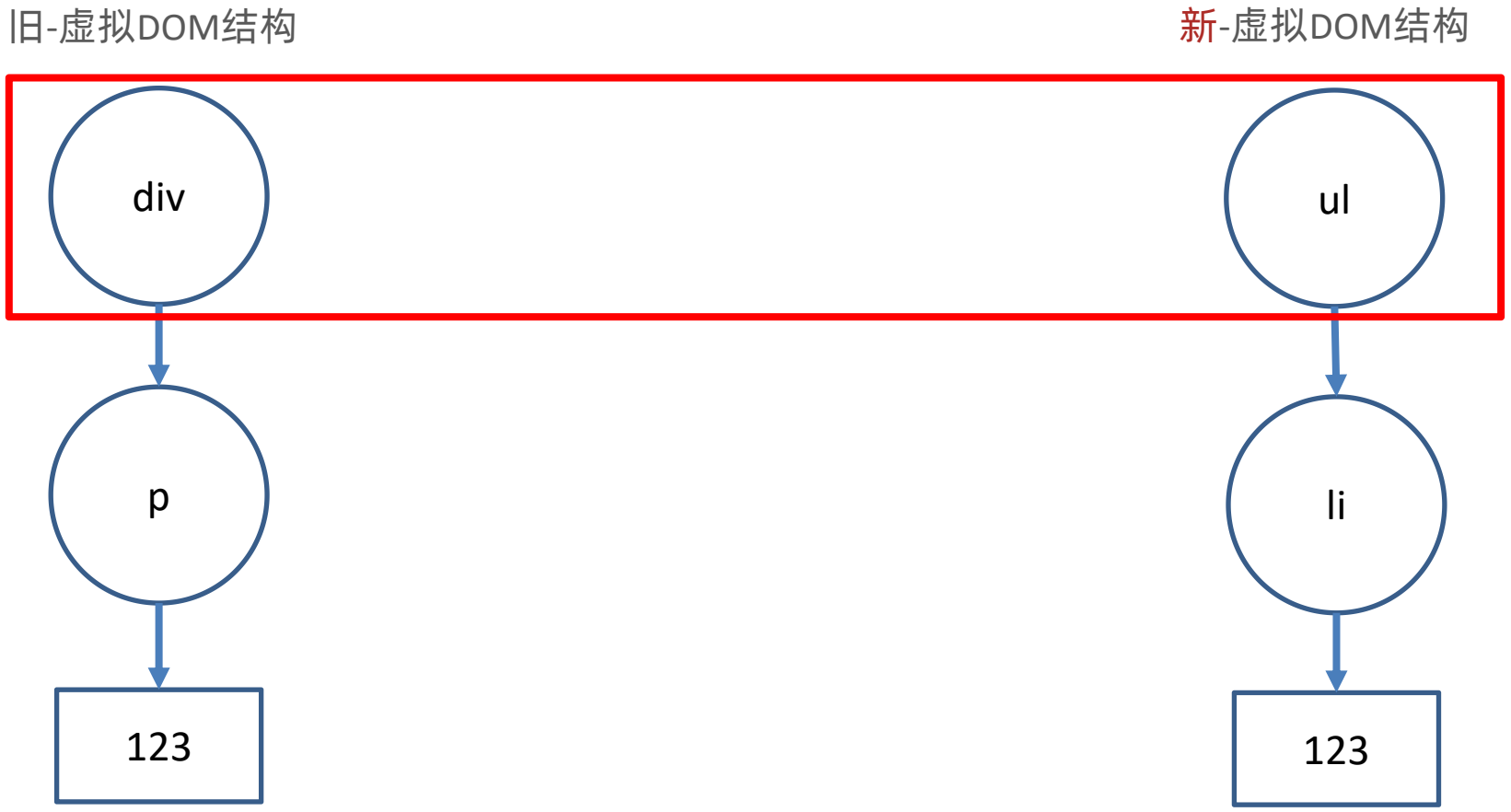
明确：通过对比 新旧虚拟dom，提高了对比性能。

但是就算是虚拟dom，和真实dom一样，也是树形结构

内部又是如何对比的呢？

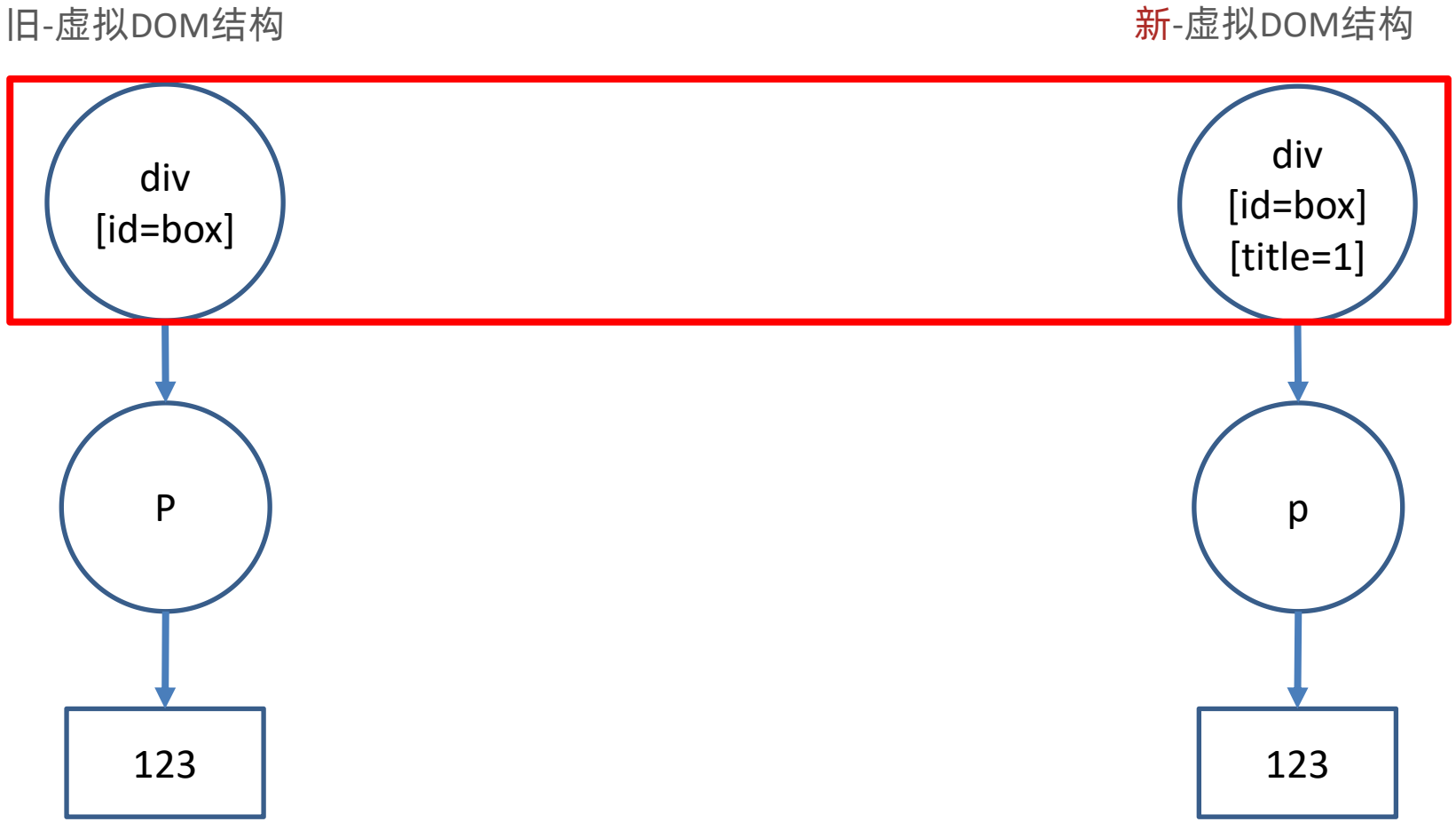
diff算法

策略1：先同层级根元素比较。 => 如果根元素变化，那么不考虑复用，整个dom树删除重建



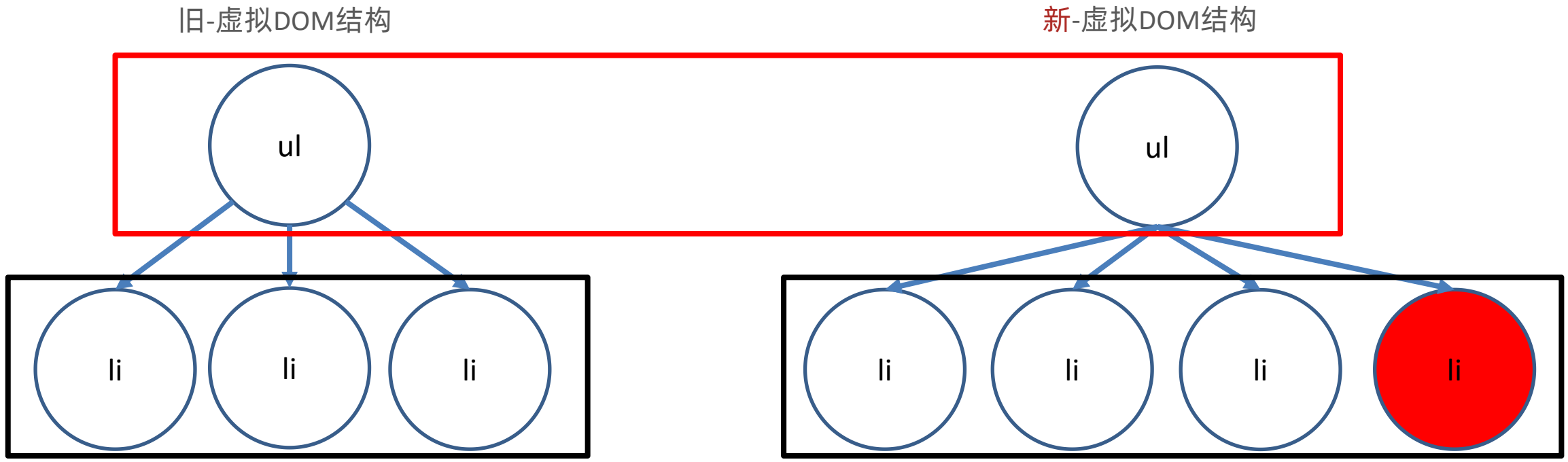
diff算法

策略1：先同层级根元素比较。 => 如果根元素不变，对比出属性的变化更新，并考虑往下递归复用。



diff算法

策略2：对比同级兄弟元素时，默认按照下标进行对比复用。



对比同级兄弟元素时，如果指定了 key，就会按照相同 key 的元素来进行对比复用。

小结

diff算法如何比较新旧虚拟DOM的呢?

1. 同层级根元素先比较

（1）如果根元素变了，删除重建dom树

（2）如果根元素没变，对比属性。并考虑往下递归复用。

2. 兄弟元素比较

（1）默认按照下标，进行对比复用

（2）如果设置了key，就会按照相同key的元素进行复用

同层级兄弟元素比较新旧变化，默认按下标比较，

如果设置了key，则优先相同key的兄弟元素比较。

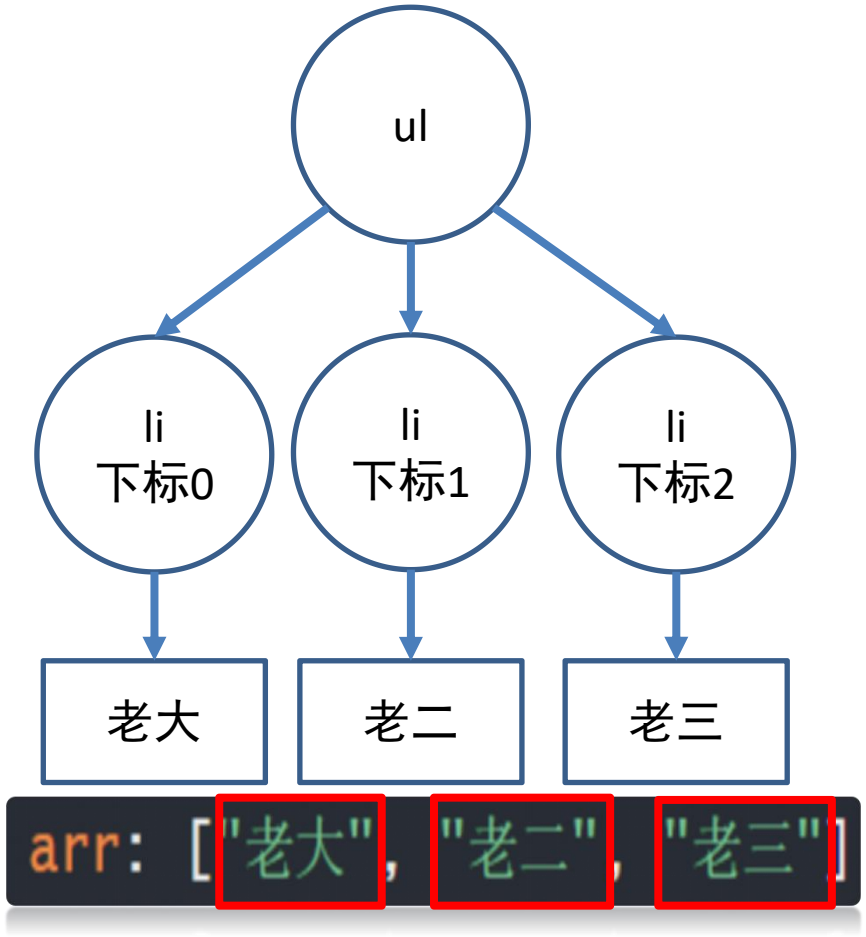
那么设置 key 和 不设置key 有什么区别呢？ 设置key有什么用呢？

key的作用 - 无 key 的情况

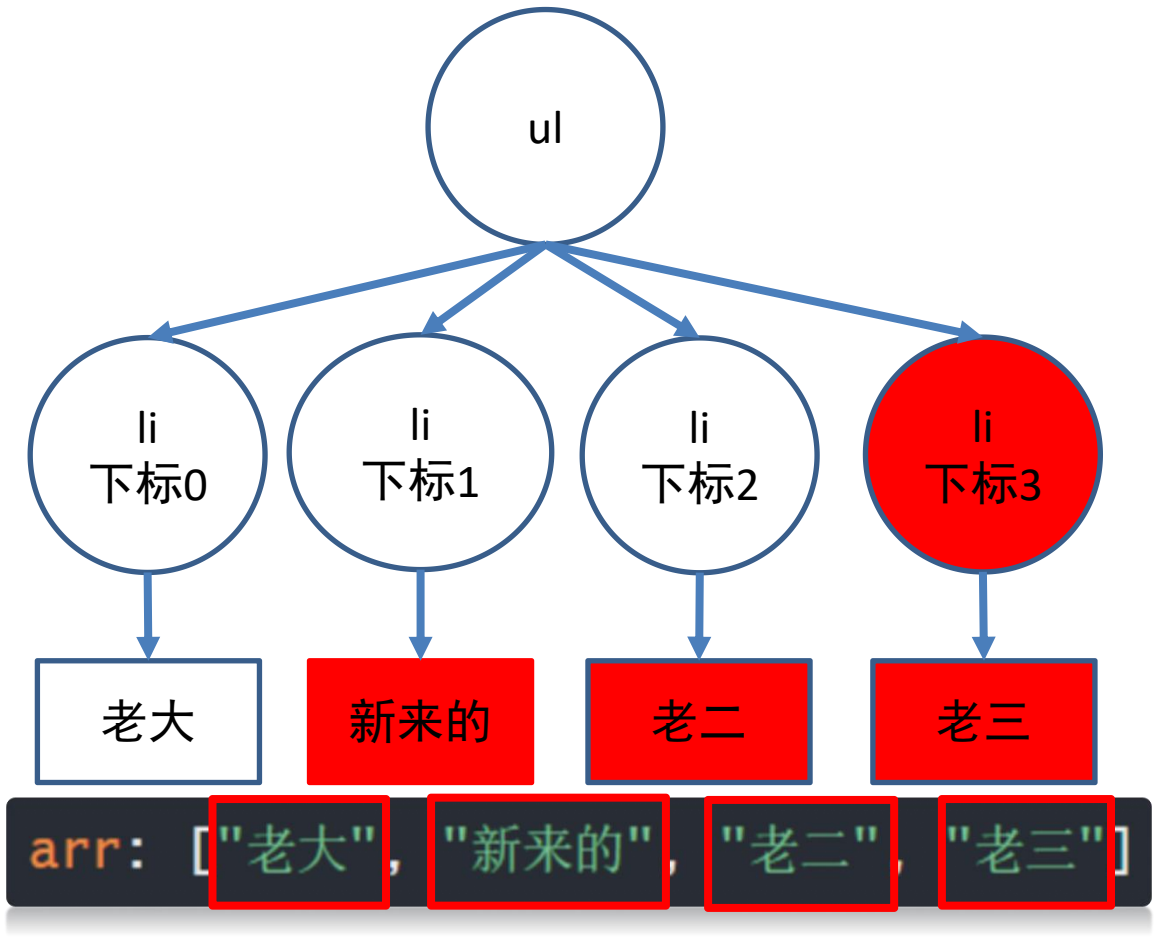
无key的情况：默认diff更新算法，是同级兄弟，按照 **下标** 对比新旧dom的差异。

key的作用 - 无 key 的情况 （按照下标对比）

旧-虚拟DOM结构



新-虚拟DOM结构

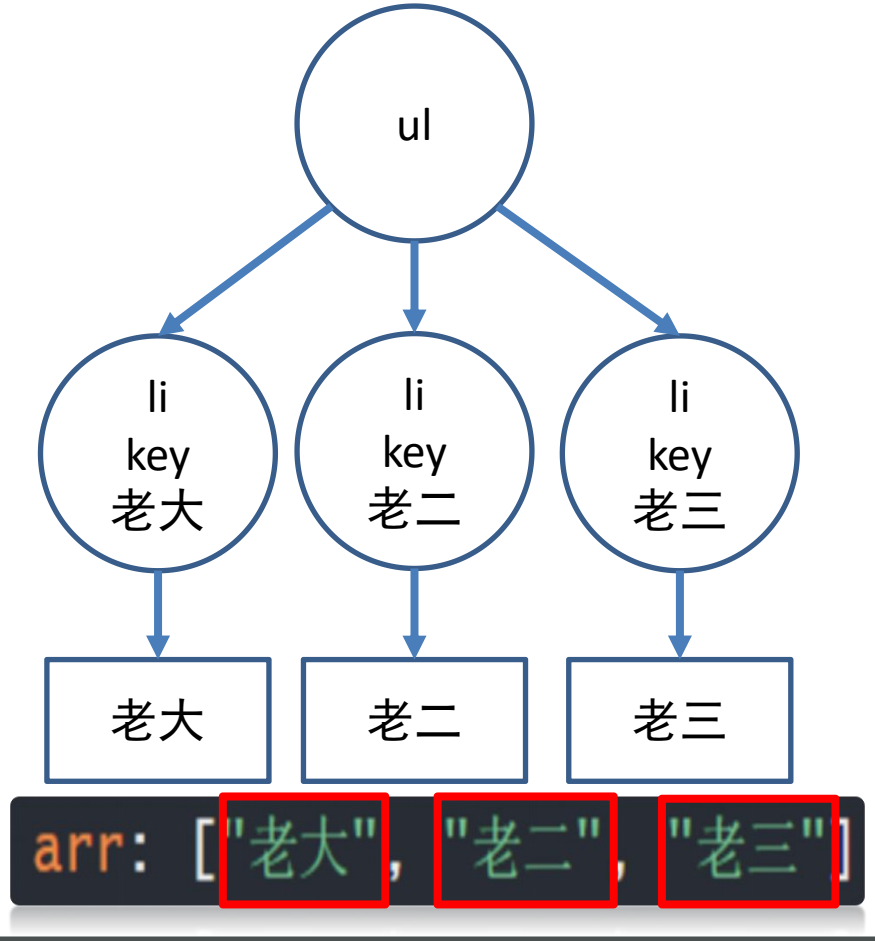


key的作用 - 有 key 的情况

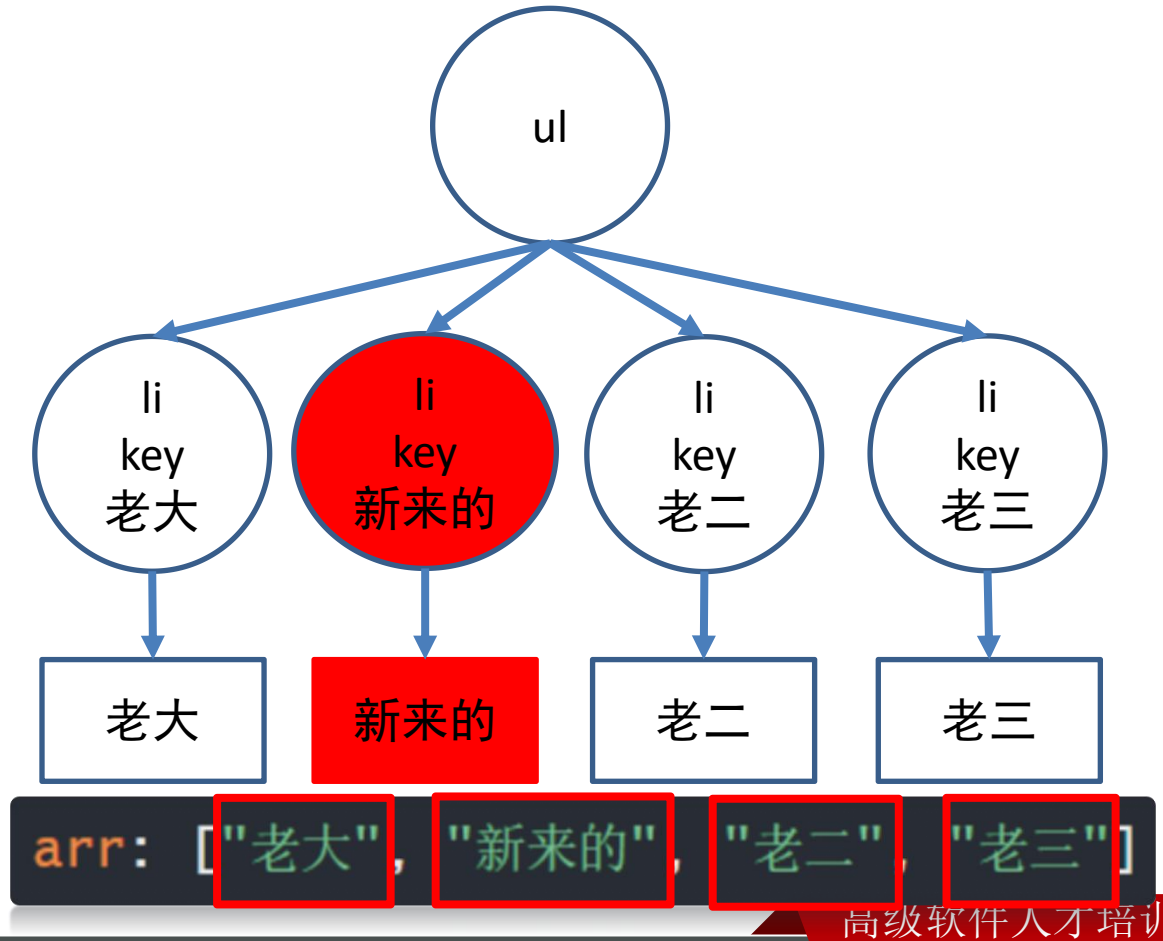
有key的情况：根据diff更新算法，同级兄弟元素，在设置了key后，会让相同key的元素进行对比。

key的要求：必须是字符串 或者 数字，且要保证唯一性！（标准的key需要指定成 id）

旧-虚拟DOM结构



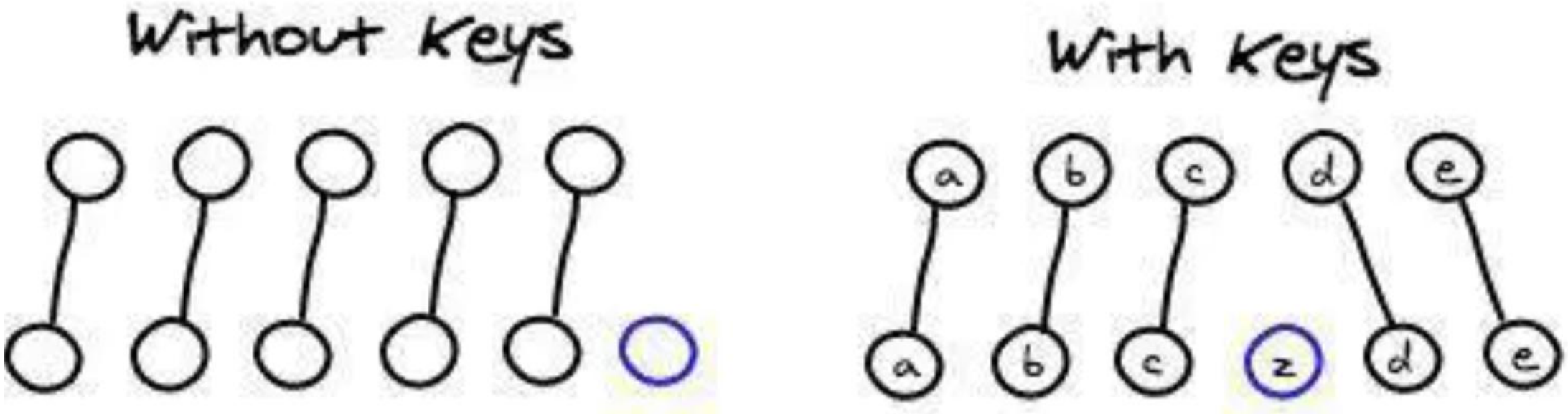
新-虚拟DOM结构



key的作用

列表循环加:key="唯一标识"，可以标识元素的唯一性，可以更好地区别各个元素。

key的作用：提高虚拟DOM的对比复用性能



小结

1. 设置 和 不设置 key 有什么区别？

不设置 key， 默认同级兄弟元素 按照下标 进行比较。

设置了key，按照 相同key 的新旧元素比较。

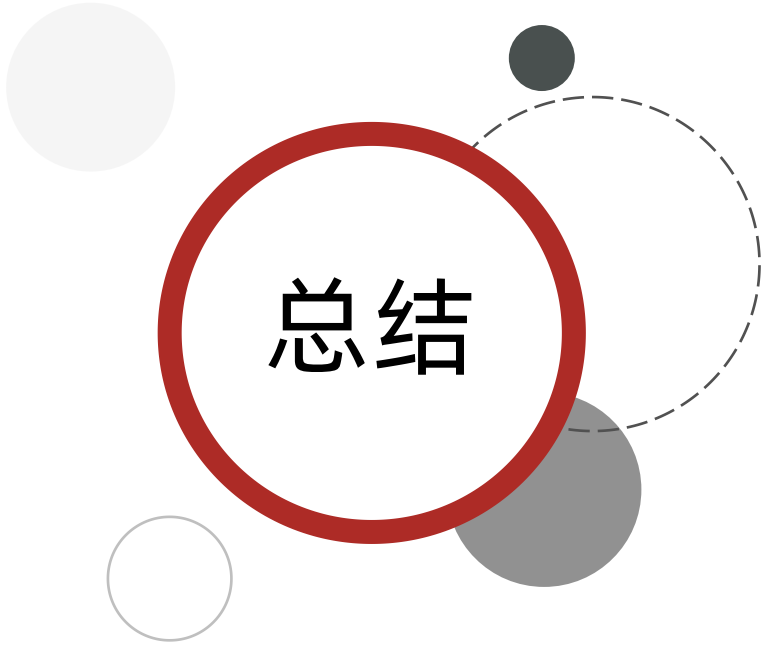
2. key值要求是？

字符串或者数值，唯一不重复

有 id 用 id，有唯一值用唯一值，实在都没有，才用索引

4. key的好处？

key的作用：提高虚拟DOM的对比复用性能



vue就地复用策略?

对比虚拟dom的差异，就地（同层级，同位置）复用结构

为什么要对比虚拟dom呢？ 什么是虚拟dom呢?

真实dom太复杂。虚拟dom就是一个描述真实dom的对象。

diff算法如何比较新旧虚拟DOM?

1 先对比根元素

根元素改变 - 删除当前DOM树重新建

根元素未变 - 对比属性 - 更新属性，并考虑向下递归对比复用

2 同级兄弟元素，对比更新：

无key - 就地按下标更新 / 有key - 按key比较

控制样式，要么操作类，要么操作行内样式，

在vue中，应该如何操作 class 类呢？如何操作style行内样式呢？

v-bind 对于class和style的增强：动态设置class 和 style

用 v-bind 动态设置标签的 class 类名

- 语法 :class="对象/数组"
- 对象：如果键值对的值为true，那么就有这个类，否则没有这个类
- 数组：数组中所有的类，都会添加到盒子上
- v-bind 对于类名操作的增强, 注意点 :class 不会影响到原来的 class 属性

用 v-bind 动态设置标签的 style 行内样式

- 语法 :style="对象/数组"

```
<div class="box" :style="{color:color, backgroundColor: bg }">123</div>
```

```
<div class="box" :style="[styleObj1, styleObj2]">123</div>
```

在vue中如何控制样式呢？

通过 :class 和 :style 动态控制样式

语法是什么呢？

:class="{类名: 布尔值}”，true使用, false不用

:class="[类名1，类名2]”

:style="{css属性名: 值}"

案例 **成绩案例 - 基本结构**

效果如下： 静态结构去md笔记 cv

编号	科目	成绩	考试时间	操作
1	语文	56	Tue Jun 07 2022 10:00:00 GMT+0800 (中国标准时间)	删除
2	数学	100	Tue Jun 07 2022 10:00:00 GMT+0800 (中国标准时间)	删除
总分：321　　平均分：80.25				

科目： 请输入科目

分数： 请输入分数

添加

案例 **成绩案例 - 列表渲染**

核心语法：

1. v-for 渲染结构

2. v-bind:class 控制样式

编号	科目	成绩	考试时间	操作
1	语文	56	Tue Jun 07 2022 10:00:00 GMT+0800 (中国标准时间)	删除
2	数学	100	Tue Jun 07 2022 10:00:00 GMT+0800 (中国标准时间)	删除
总分：321　　平均分：80.25				

科目：请输入科目

分数：请输入分数

添加

案例 **成绩案例 - 删除功能**

删除思路：

1. 注册点击事件，传递参数，阻止默认行为
2. 在method中提供对应函数
3. 根据id删除对应项

编号	科目	成绩	考试时间	操作
1	数学	100	2022-06-05 06:44:51	删除
2	体育	30	2022-06-05 07:41:01	删除
总分：130　　平均分：65.00				

案例

成绩案例 - 新增功能

添加思路：

1. 获取科目 和 分数
2. 给添加按钮注册点击事件
3. 给list数组添加一个对象
4. 重置表单数据

编号	科目	成绩	考试时间	操作
1	数学	100	2022-06-05 06:44:51	删除
2	体育	30	2022-06-05 07:41:01	删除
总分：130　　平均分：65.00				

科目：请输入科目

分数：请输入分数

添加

案例

成绩案例 - 处理日期格式

需求: 表格里的时间用封装函数 + moment模块, 格式化成YYYY-MM-DD HH:mm:ss 格式

效果如下:

编号	科目	成绩	考试时间	操作
1	数学	100	2022-06-05 06:44:51	删除
2	体育	30	2022-06-05 07:41:01	删除
总分：130　　平均分：65.00				

目录

Contents

计算属性的基本使用

计算属性: 一个特殊属性, 值依赖于另外一些数据动态计算出来。

注意:

- 1. 计算属性必须定义在 computed 节点中
- 2. 计算属性必须是一个 function,计算属性必须有返回值
- 3. 计算属性不能被当作方法调用, 要作为属性来用

```
computed: {
    "计算属性名" () {
        return "值"
    }
}
```

```
<template>
  <div>
    <p>和为: {{ num }}</p>
  </div>
</template>
```

计算属性也是属性, 所以不要和data里重名, 用起来也和data类似

小结

1. 什么是计算属性?

一个特殊属性, 值依赖于另外一些数据动态计算出来。

2. 计算属性特点?

函数内使用的变量改变, 重新计算结果返回

3. 计算属性注意事项?

计算属性名和data里名字不能重复

问题1：不用计算属性，用函数调用能实现吗？

问题2：计算属性优势在哪里？

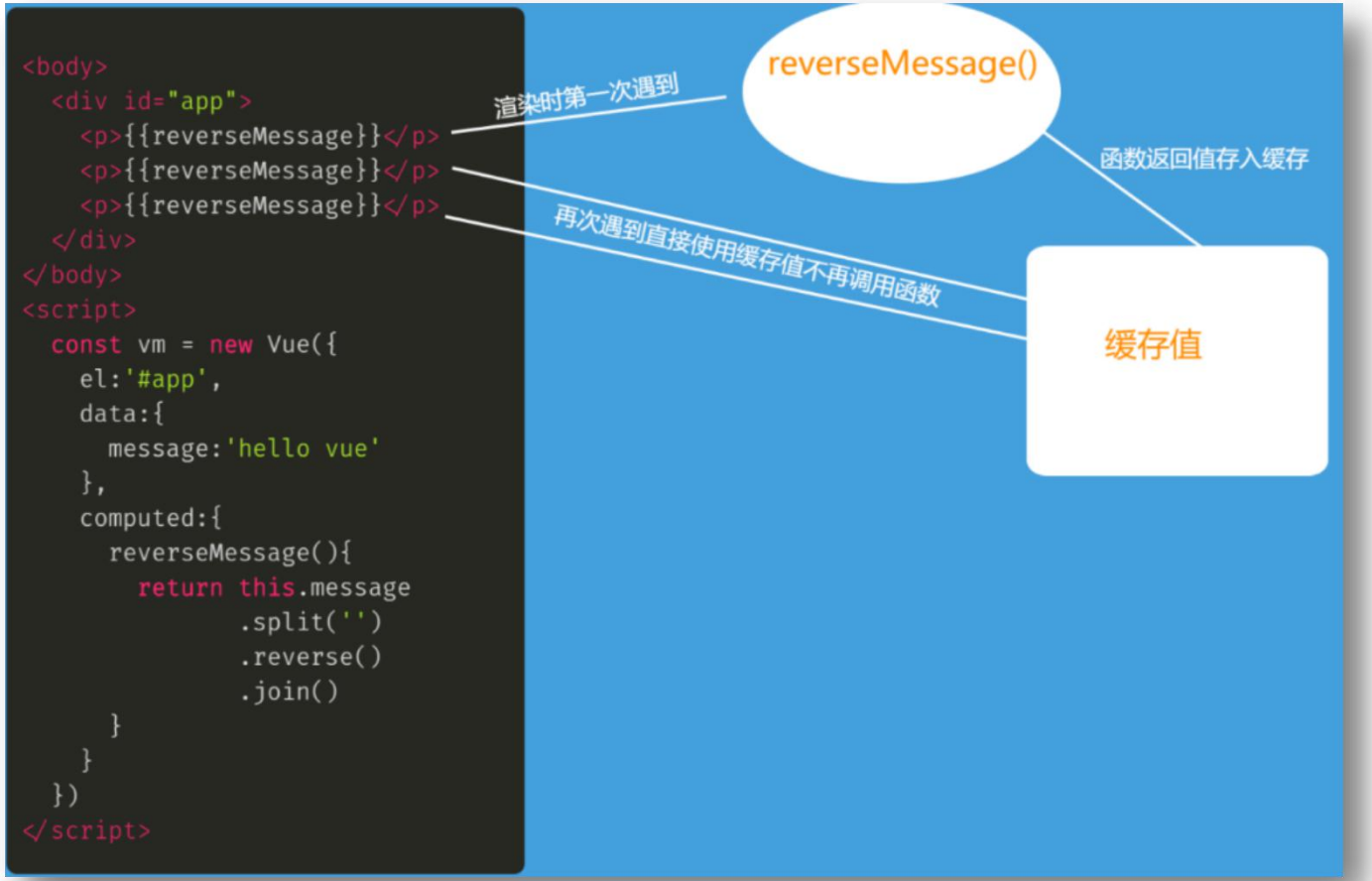
计算属性的缓存说明

计算属性，基于依赖项的值进行缓存，依赖的变量不变，都直接从缓存取结果。

计算属性如果被多次使用，性能极高

```
<template>
  <div>
    <p>{{ reverseMessage }}</p>
    <p>{{ reverseMessage }}</p>
    <p>{{ reverseMessage }}</p>
    <p>{{ getMessage() }}</p>
    <p>{{ getMessage() }}</p>
    <p>{{ getMessage() }}</p>
  </div>
</template>
```

```
<body>
  <div id="app">
    <p>{{reverseMessage}}</p>
    <p>{{reverseMessage}}</p>
    <p>{{reverseMessage}}</p>
  </div>
</body>
<script>
  const vm = new Vue({
    el:'#app',
    data:{
      message:'hello vue'
    },
    computed:{
      reverseMessage(){
        return this.message
                .split('')
                .reverse()
                .join()
      }
    }
  })
</script>
```



小结

计算属性特点？

1 计算完了一次，就会自动进行缓存

2 如果依赖项不变，下次使用直接从缓存取

3 直到依赖项改变，函数才会重新执行并重新缓存

多次使用计算属性，性能会极高

案例

完成总成绩和平均成绩的显示

效果如下：

编号	科目	成绩	考试时间	操作
1	数学	100	2022-06-05 06:44:51	删除
2	体育	30	2022-06-05 07:41:01	删除
总分：130　　平均分：65.00				

计算属性的设置问题-完整写法

1. 计算属性默认情况下只能获取，不能修改。

2. 要给计算属性赋值，就必须写计算属性的完整写法！

```
computed: {
  full: {
    get () {
      ...
    },
    set (value) {
      ...
    }
  }
}
```

什么时候用计算属性完整写法?

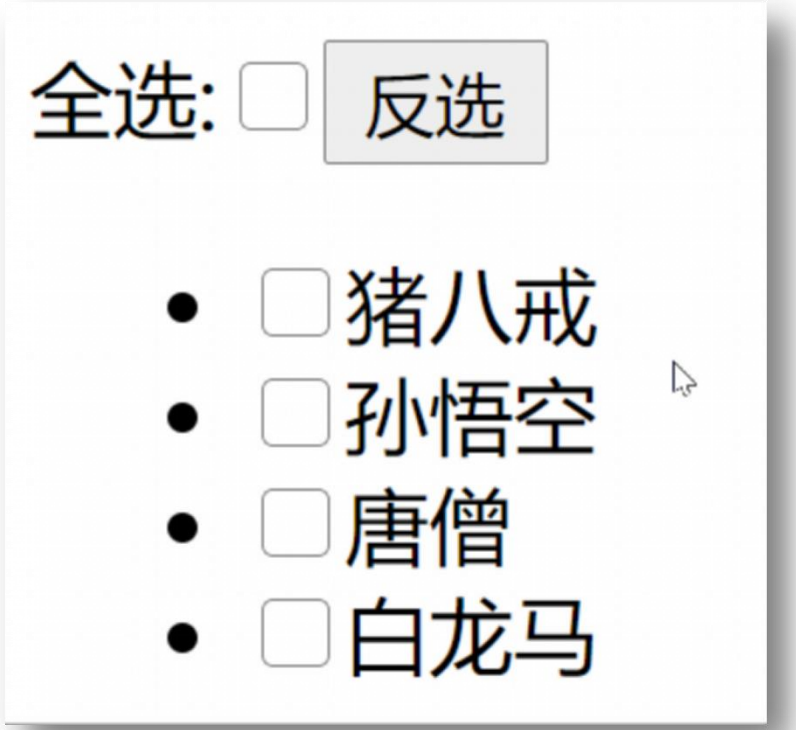
给计算属性变量赋值时

案例

全选反选案例

需求: 小选框都选中(手选), 全选自动选中

效果演示:



目录

Contents

vue中想监听数据的变化，应该怎么办？

侦听器 watch

watch: 可以侦听到 data/computed 属性值的改变

语法:

```
watch: {
    "被侦听的属性名" (newVal, oldVal){

    }
}
```

```
<template>
  <div>
    <input type="text" v-model="name">
  </div>
</template>

export default {
  data(){
    return {
      name: ""
    }
  },
  watch: {
    name(newVal, oldVal){ // 当msg变量的值改变触发此函数
      console.log(newVal, oldVal);
    }
  }
}
```

小结

如何侦听到某个变量值改变呢?

使用watch配置项, key是要侦听的data/计算属性名

侦听器 深度侦听和立即执行

如果监听的是复杂数据类型，需要深度监听，需要指定deep为true, 需要用到监听的完整的写法

语法：

```
watch: {
    "要侦听的属性名": {
        immediate: true, // 立即执行
        deep: true, // 深度侦听复杂类型内变化
        handler (newVal, oldVal) {

        }
    }
}
```

```
<template>
  <div>
    <input type="text" v-model="user.name">
    <input type="text" v-model="user.age">
  </div>
</template>
```

小结

1. 如何侦听一个对象/数组呢?

   把侦听器写成对象形式, 给handler方法和deep:true

2. 如何让侦听器函数马上执行一次呢?

   immediate: true

案例

监听list变化，同步到本地

需求：把数据实时同步到本地缓存

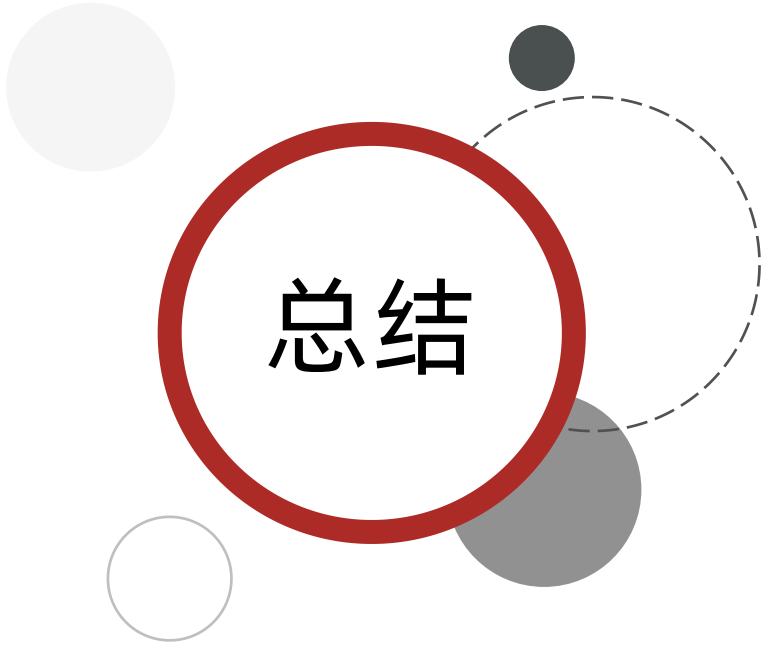
分析：

① 在watch侦听list变化的时候，把最新的数组list转成JSON字符串存入到localStorage本地

② data里默认把list变量从本地取值，如果取不到给个默认的空数组

效果：

新增/删除 - 刷新页面 - 数据还在



1. key作用是什么
2. 虚拟dom是什么，为什么要有虚拟dom， diff算法
3. 动态设置class或style
4. 什么是Vue计算属性
5. Vue侦听器的作用

1. 把今日课上的练习和案例来一遍(可以边看视频边写，然后自己整理思路，从0来一遍)
2. 预习下一天内容