# vue生命周期

beforeCreate（创建前） 在数据观测和初始化事件还未开始

created（创建后） 完成数据观测，属性和方法的运算，初始化事件，$el属性还没有显示出来

beforeMount（载入前） 在挂载开始之前被调用，相关的render函数首次被调用。实例已完成以下的配置：编译模板，把data里面的数据和模板生成html。注意此时还没有挂载html到页面上。

mounted（载入后） 在el 被新创建的 vm.$el 替换，并挂载到实例上去之后调用。实例已完成以下的配置：用上面编译好的html内容替换el属性指向的DOM对象。完成模板中的html渲染到html页面中。此过程中进行ajax交互。

beforeUpdate（更新前） 在数据更新之前调用，发生在虚拟DOM重新渲染和打补丁之前。可以在该钩子中进一步地更改状态，不会触发附加的重渲染过程。

updated（更新后） 在由于数据更改导致的虚拟DOM重新渲染和打补丁之后调用。调用时，组件DOM已经更新，所以可以执行依赖于DOM的操作。然而在大多数情况下，应该避免在此期间更改状态，因为这可能会导致更新无限循环。该钩子在服务器端渲染期间不被调用。

beforeDestroy（销毁前） 在实例销毁之前调用。实例仍然完全可用。

destroyed（销毁后） 在实例销毁之后调用。调用后，所有的事件监听器会被移除，所有的子实例也会被销毁。该钩子在服务器端渲染期间不被调用。

keep-alive是Vue提供的一个抽象组件，用来对组件进行缓存，从而节省性能，

当组件在keep-alive内被切换时组件的activated、deactivated这两个生命周期钩子函数会被执行

# vue实现双向绑定的原理

在创建Vue实例的时候给传入的data进行数据劫持，同时视图编译的时候，对于使用到data中数据的地方进行创建Watcher对象，然后在数据劫持的getter中添加到发布者对象中，当劫持到数据发生变化的时候，就通过发布订阅模式以回调函数的方式通知所有观察者操作DOM进行更新，从而实现数据的双向绑定。

# vue双向绑定原理，getter，setter

Vue实现数据双向绑定的原理：Object.defineProperty（）

vue实现数据双向绑定主要是：采用数据劫持结合发布者-订阅者模式的方式，通过Object.defineProperty（）来劫持各个属性的setter，getter，在数据变动时发布消息给订阅者，触发相应监听回调。当把一个普通 Javascript 对象传给 Vue 实例来作为它的 data 选项时，Vue 将遍历它的属性，用 Object.defineProperty 将它们转为 getter/setter。用户看不到 getter/setter，但是在内部它们让 Vue 追踪依赖，在属性被访问和修改时通知变化。

# vuex原理

在 Vuex 中，$store 被直接注入到了组件实例中，因此可以比较灵活的使用：

使用 dispatch 和 commit 提交更新

通过 mapState 或者直接通过 this.$store 来读取数据

Vuex其实和Vue的原理一样，是通过 getter和setter来比较的

6.vue组件data为什么用return

组件是一个可复用的实例，当你引用一个组件的时候，组件里的data是一个对象，所有用到这个组件的都引用的同一个对象，就会造成数据污染。

将data封装成函数后，在引用组件的时候，我们只是调用了data函数生成的数据副本，是一个新对象，避免了数据污染。

# vue中计算属性和方法的区别

methods计算方法：当修改其中一个属性时，其他属性的值都没改变，但methods里的方法都被执行

computed计算属性：会发现当修改其中一个值的时候，只会执行于其相关的方法。

watch 属性。当你有一些数据需要随着其它数据变动而变动时，使用watch属性