**停止编译（生命周期停止），直到vue实例调用vm.$mount(el)**

**把data的数据和模板生成html**

**vue内部初始化事件**

**开始监控data对象数据变化**

**将外部HTML作为模板编译**

**是否存在template参数**

**给vue实例添加$el成员，将编译好的模板渲染到真实DOM中**

**将模板编译成render函数**

**是否存在el，便于下步操作中如果没有template，好找到对应的HTML**



1. **开始（第一次页面加载时触发）**
2. beforeCreate：事件、数据和方法初始化（组件实例刚被创建，组件属性计算之前，如data属性等）。--》**loading事件在加载实例时触发**
3. created：数据绑定（组件实例创建完成，**data属性已绑定（初始化完成）**，但DOM还未生成，$el属性还不存在）。--》**结束loading事件，异步请求调用**
4. beforeMount：看le和template编译那个模板（模板编译/挂载之前），完成了 el 和 data 初始化—》**获取到DOM节点**
5. mounted：把编译好的模板渲染到真实的DOM节点中（模板编译/挂载之后）--》**对数据统一处理，在这里写上相应函数**
6. **数据发生改变**
   1. beforeUpdate：组件更新之前（只有和模版中的数据绑定了才会发生更新）
   2. updated：组件更新之后（只有和模版中的数据绑定了才会发生更新）
7. **销毁**
   1. beforeDestroy：组件销毁前调用：
   2. destroyed：组件销毁后调用：调用后，Vue 实例指示的所有东西都会解绑定，所有的事件监听器会被移除，所有的子实例也会被销毁。重新改变值，没有响应（不再受vue控制了）。但是原先生成的dom元素还存在

