Vue实例方法

// 处理 $data $props $set $del $watch stateMixin(Vue)

- 处理 data 数据，定义 get 方法， 支持通过 this._data 来访问
- 处理 props 数据，定义 get 方法， 支持通过 this._props 来访问
- 若直接给 this._data 赋值，则提示不能直接给 data 赋值
- 将 props 设置为只读，当你给 props 赋值时，会提示你 props 时只读的。
- 将 $data $props 挂载到 Vue 原型链上
- 将 $set $del 挂载到 Vue 原型上
- 将 $watch 挂载到原型链上

// $on $emit $once $off eventsMixin(Vue)

- 例子：this.$on('custom-event',function(){})
  将所有的事件和对应的回调函数放在 vm._events 对象上 格式为
  { 'csutom-event':[cb1,cb2,..] }
  $on
  - 事件为数组的情况
    循环遍历 事件数组在执行 $on 方法
  - 不是数组的情况
    判断在 _events 的对象中是否存在事件，若果存在将回调放入事件对应的事件数组中
    例子：
    {
    ' custom-events ':['cb1']
    }
  - 如果发现事件名称匹配上 hook 则将 _hasHookEvent 置为 true
- 先通过 $on 添加事件，然后在事件回调函数中先调用 $off 移除事件，再执行用户的回调
  $once
  - 自定义一个 on 方法
    (先调用 vm.$off 方法 在执行回调函数)
  - 先用 $on 将事件添加到 vm._events 里，在触发事件时，先移除监听事件在执行回调
    将包装后的函数作为回调函数
    执行 $on 方法
- 移除 vm._events 的指定事件的指定回调函数的
  1. 没有参数 移除所有事件
  2. 没有 fn 参数 移除指定事件
  3. 有 event 和 fn 移除指定事件的指定回调函数
     操作通过 $on 设置的 vm.$events
     $off
  - 如果事件为空 移除所有的事件监听器
    将 vm._events = Object.create(null)
  - 如果是一个数组，则遍历递归调用 $off
  - 如果是单个事件，且回调函数为空，则将该事件的所有回调函数全部移除
  - 如果是单个事件，且回调函数不为空，则将指定事件的指定回调函数移除。
- 触发实例上的指定事件，vm._event[event] => cbs => loop cbs => cb(args)
  $emit
  - 从 vm._events 获取指定的事件，将回调类数组转化为数组，遍历数组执行回调函数。

// _update $foreUpdate $destory lifecycleMixin(Vue)

- // 负责更新页面，页面首次渲染和后续更新的入口位置，也是 patch 的入口位置
  _update
  - 获取 vue 实例
    获取根节点
  - 如果根节点不存在【初次渲染】
    vm.$el = vm.__patch__(vm.$el, vnode, hydrating, false /* removeOnly */)
  - 如果根节点存在
    // updates
    vm.$el = vm.__patch__(prevVnode, vnode)
- // 强制更新当前组件以及子组件，并非所有的组件
  $foreUpdate
  - 判断实例下的 _watchers 是否存在，存在则遍历调用 update 方法
- $destory
  - 判断组件是否开始销毁，如果已经销毁则结束执行
  - hook 函数 beforeDestory
  - 将销毁属性置为 true
  - 将组件本身从父组件的 children 属性中移除
  - 移除 watcher
  - 将自己标记为已销毁
  - 更新界面
  - hook 函数 destoryed
  - 移除当前组件所有事件监听
  - 将 vnode 置为 null 断开与父组件的联系

// $nextTick _render renderHelper renderMixin(Vue)

- // 在组件实例上挂载一些运行时用到的实例方法
  installRenderHelpers(Vue.prototype)
- //获取 render 函数
  //用户实例化 Vue 提供的 render 函数
  //编译器编译生成的 render 函数
  //执行组件的 render 得到 vnode
  _render
  - 从 options 获取 render 和 _parentVNode
  - 若 _parentVNode 存在则标准化 slot 相关属性
  - 执行 render 函数 得到 vnode
  - 若有多个根节点则抛出警告，有多个根节点，并将空的 vnode 返回
- $nextTick 就是前文的 nextTick