1. **原理**

数据驱动视图 UI = render(state)

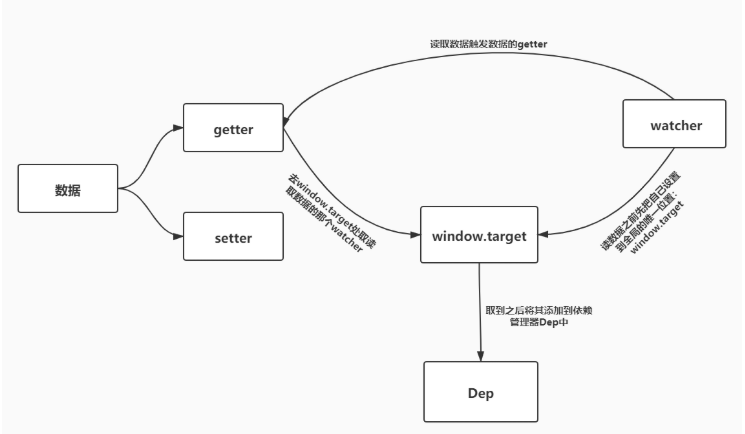
1. **object数据变化侦测**

附注：Angular：脏值检测；react：对比虚拟Dom

1. Object的监测：

object“可监测”：js中的Object.defineProperty()方法，里面提供了get()、set()方法，可以在值变化时触发。还定义了observe类，使得操作不会重复。

1. 依赖收集：“谁用到谁更新”，为每一个数据建立一个依赖数组（后期使用依赖管理器dep类）。所以，就是在getter中，建立这个依赖关系，在setter中触发（通知）这个变化（更新）
2. 依赖是什么？watcher类的实例，就是这个依赖。数据变化，通知依赖对应的watcher实例去通知视图更新。



1. Array数据变化侦测
2. 虚拟dom
3. Dom-diff算法
4. **子节点的更新与优化**

**更新**：

VNode上的子节点数组记为newChildren，把旧的oldVNode上的子节点数组记为oldChildren，把newChildren里面的元素与oldChildren里的元素一一进行对比，对比两个子节点数组肯定是要通过循环，外层循环newChildren数组，内层循环oldChildren数组，每循环外层newChildren数组里的一个子节点，就去内层oldChildren数组里找看有没有与之相同的子节点

**优化**：

先对比特殊位置的子节点后（新前与旧前，新后与旧后，新后与旧前，新前与旧后），如若都不相同，再用原始的层层对比。

1. 模板编译解析
2. 生命周期