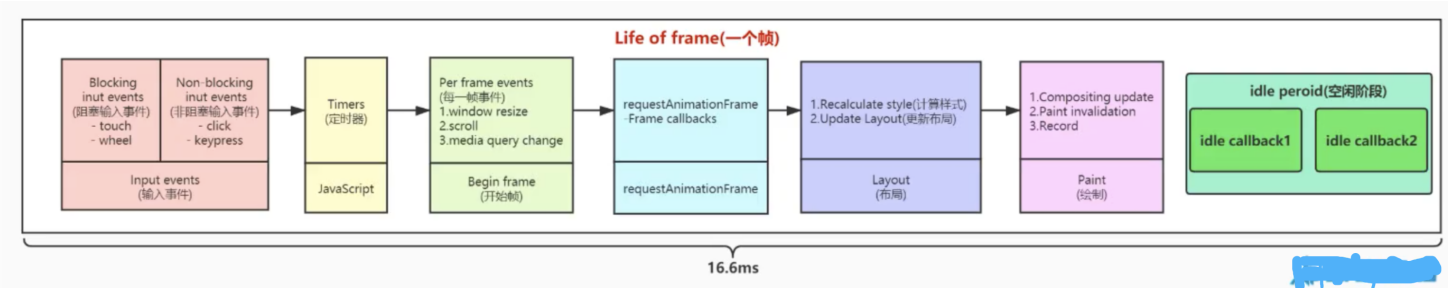
Fiber

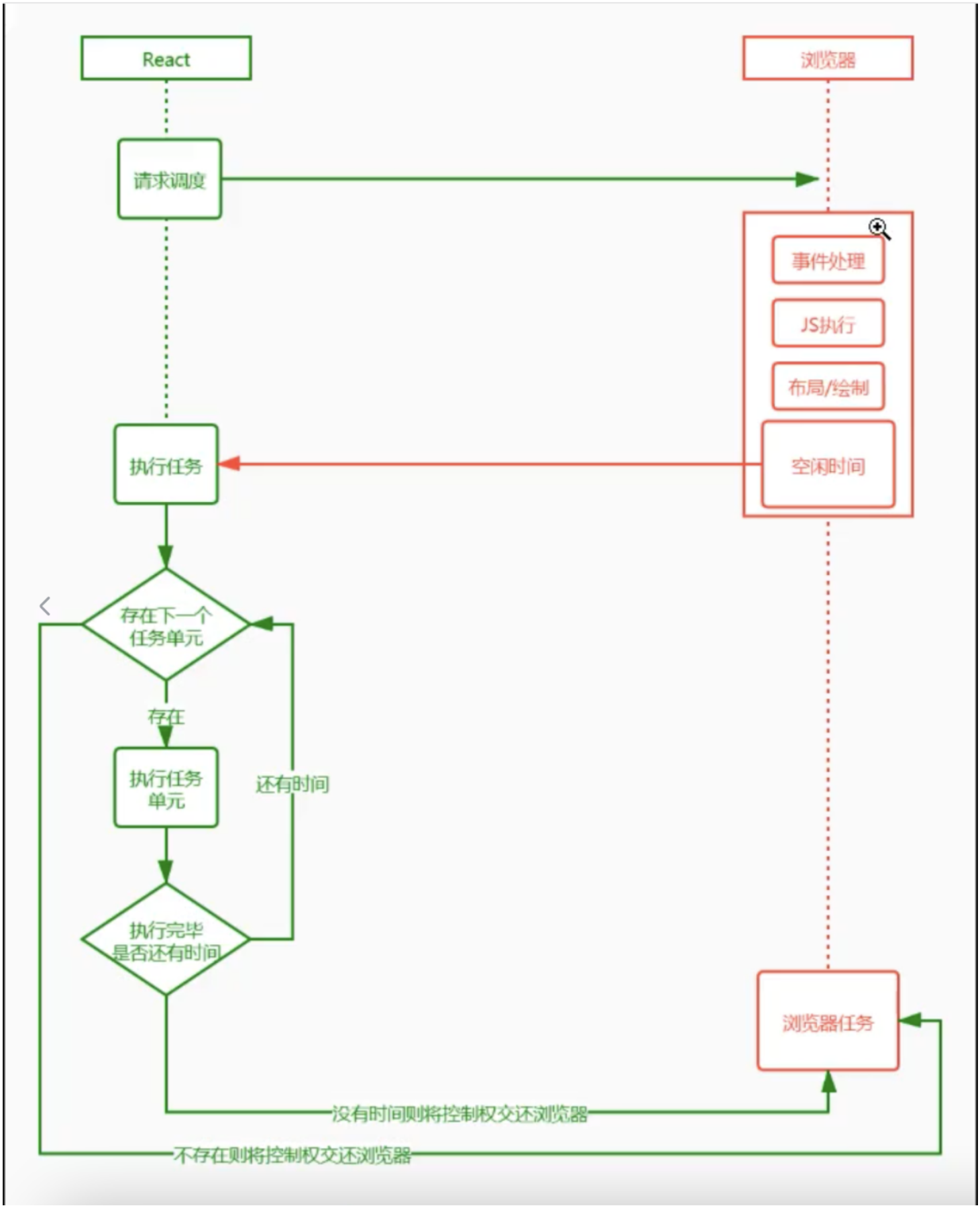
1. 帧

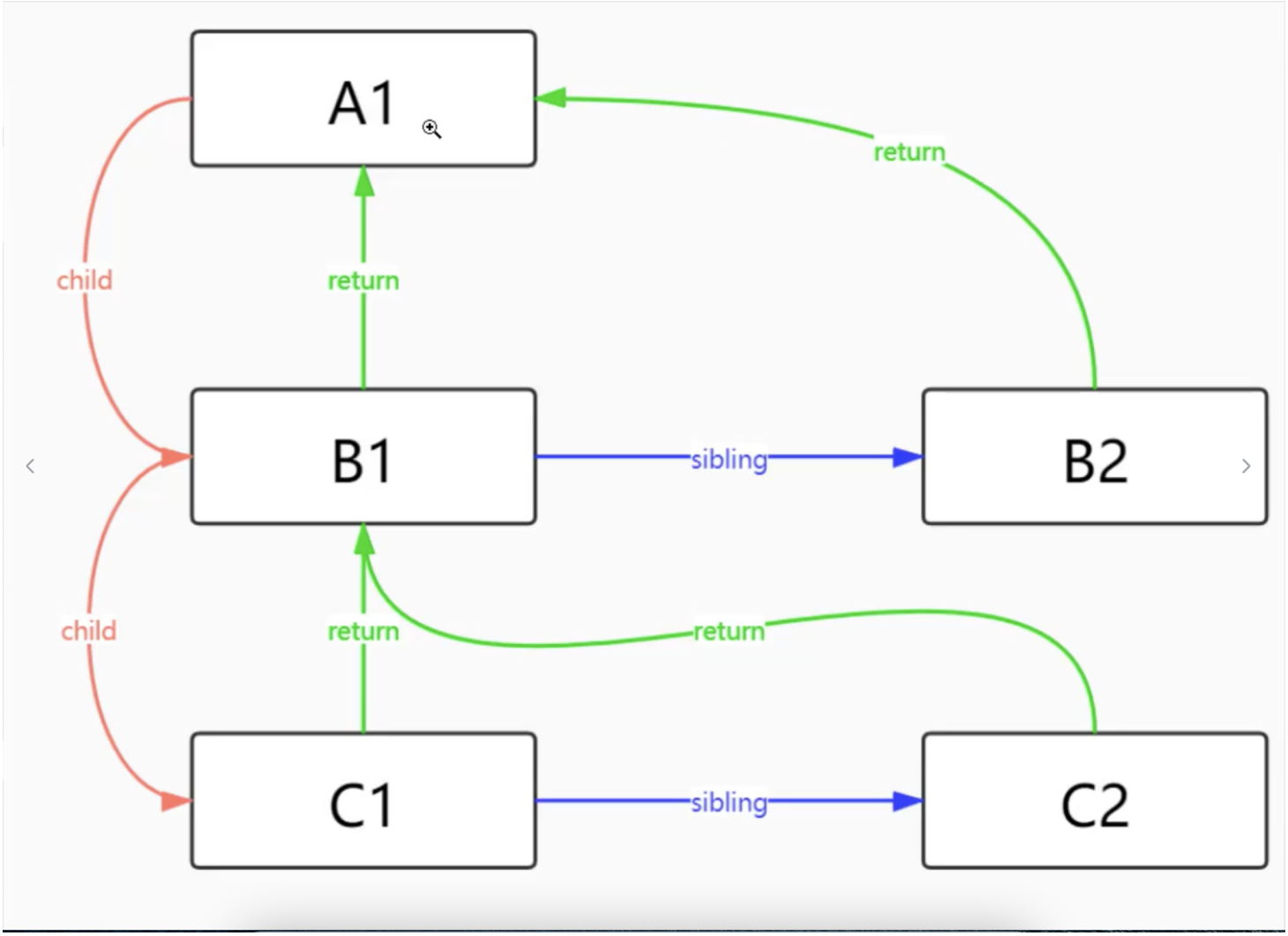
- 目前大多数设备的屏幕刷新率为60次/秒
- 当每次绘制的帧数（FPS）达到60时，页面是流畅的，小于这个值时，用户会感觉到卡顿
- 每个帧的预算时间是16.66毫秒（1秒/60）
- 每个帧的开头包括样式计算、布局和绘制
- javascript执行JavaScript引擎和页面渲染引擎在同一个渲染线程，GUI渲染和JavaScript执行两者是互斥的
- 如果某个任务执行时间过长，浏览器会推迟渲染



2. 什么是fiber

- 我们可以通过某些调度策略合理分配CPU资源，从而提高用户的响应速度
- 通过 Fiber 架构，让自己的协调过长变成可被中断。适度低让出CPU执行权，以便可以让浏览器及时响应用户的交互


React
浏览器
请求调度
事件处理
JS执行
布局/绘制
空闲时间
执行任务
存在下一个
任务单元
存在
执行任务
单元
还有时间
执行完毕
是否还有时间
浏览器任务
没有时间则将控制权交还浏览器
不存在则将控制权交还浏览器



3.rAF

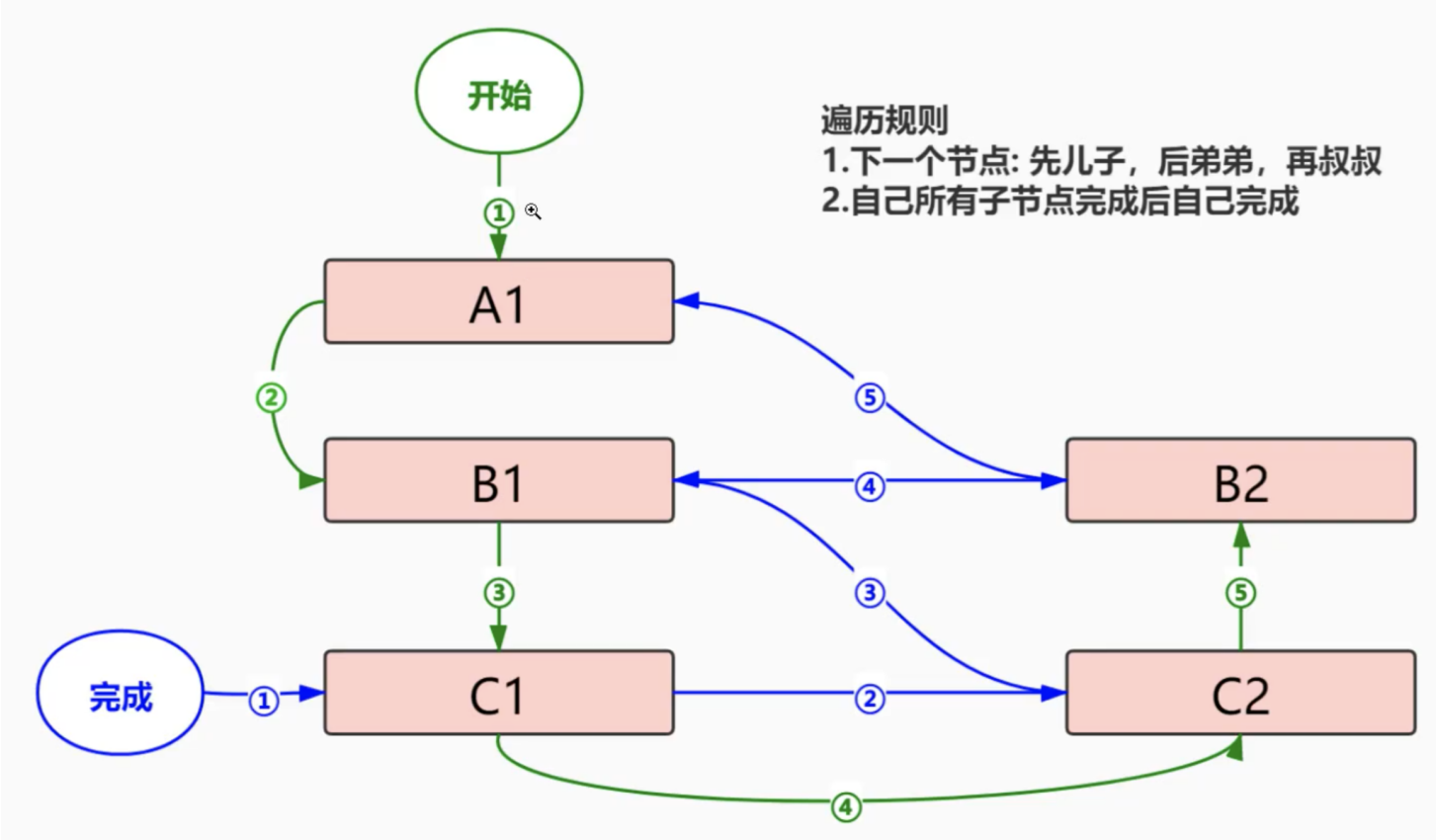
- requestAnimationFrame回调函数会在绘制之前执行
- requestAnimationFrame(callback) 会在浏览器每次重绘前执行callback回调，每次callback执行的时机都是浏览器刷新下一帧渲染周期的起始点
- requestAnimationFrame(callback) 的回调callback回调参数timestamp是回调被调用的时间，也即是当前帧的起始时间
- rAfTime performance.timing.navigationStart + performance.now() 约等于 Date.now()

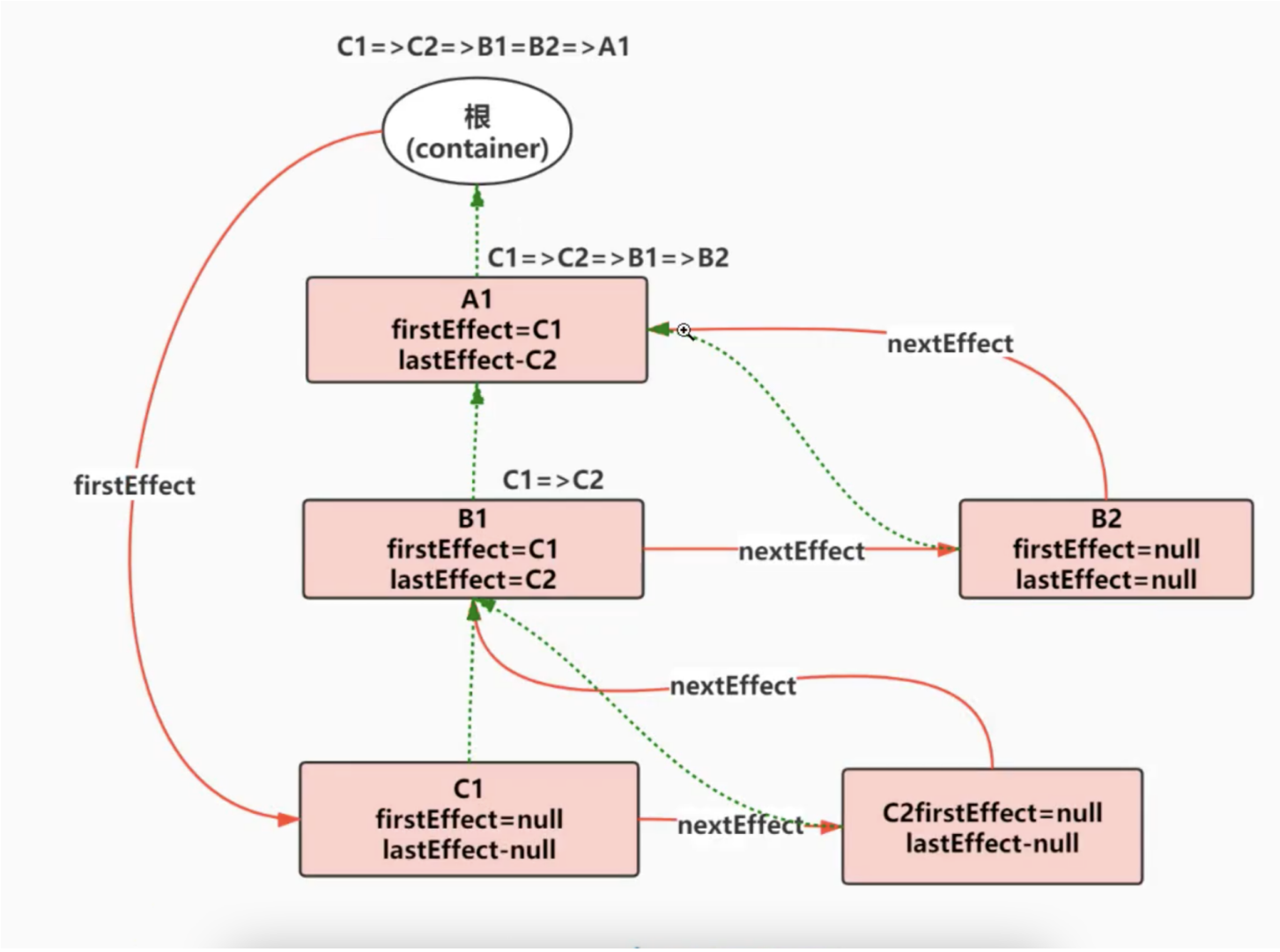
4.Fiber执行阶段

- 每次渲染有两个阶段： Reconciliation(协调render阶段)和Commit(提交阶段)
- 协调阶段： 可以认为是diff阶段，这个阶段可以被终端，这个阶段会找出所有节点变更，例如节点新增、删除、属性变更等等，这些变更React称之位副作用（Effect）
- 提交阶段： 将上个阶段计算出来的需要处理的副作用（Effects）一次性执行了。这个阶段必须同步执行，不能被打断

5.render阶段

- 从定点开始遍历
- 如果有第一个儿子，先遍历第一个儿子
- 如果没有第一个儿子，标志着此节点遍历完成
- 如果有弟弟遍历弟弟
- 如果没有下一个弟弟，返回父节点完成父节点遍历，如果有叔叔遍历叔叔
- 没有父节点遍历结束




C1=>C2=>B1=B2=>A1
根
(container)
C1=>C2=>B1=>B2
A1
firstEffect=C1
lastEffect-C2
nextEffect
firstEffect
C1=>C2
B1
firstEffect=C1
lastEffect=C2
nextEffect
B2
firstEffect=null
lastEffect=null
nextEffect
C1
firstEffect=null
lastEffect-null
nextEffect
C2firstEffect=null
lastEffect-null