UIKit

1. 1.uikit中的视图树和控制器树
2. iOS中的事件传递和事件响应

iOS系统处理用户事件分这么几个阶段：

* 1. APP唤醒和事件分发

用户和iOS设备的交互，如触摸，经屏幕捕捉交给IOKit，IOKit通过machport的方式将事件传递给前台进程，UIKit在主线程runloop注册的source1监听该machport，主线程runloop被唤醒后，处理该source1，uikit在该source1的回调中，将事件包装为source0添加到runloop中，在合适的时机处理这些source0，source0的回调会基于APP的视图树，调用根视图uiwindow的sendevent，sendevent内部分两个部分，一个是利用hittest:event方法寻找响应视图，一个是找到视图后，进行事件响应

* 1. 事件传递

简而言之事件传递就是在视图树上递归调用hittest：event：方法寻找最顶层的可响应视图，简单来讲，事件传递就是在视图树上进行DFS+剪枝寻找可响应的最顶层视图。

hittest:event：函数是uiview默认实现的函数，默认实现是：

- (void)hitTest:(CGPoint)point event:(UIEvent)event {

if(self.userInteractionEnable == NO || self.hidden == YES || self.alpha <= 0.01) return nil;

if([self pointInside:point event:event] == NO) return nil;

for (UIView \*subView in self.subviewsReverseObjectEnumator){

CGPoint convertPoint = [subView convertPoint:point fromView:self];

UIView \*hitView = [subView hitTest:convertPoint event:event];

if (hitView) return hitView;

}

return self;

}

* 1. 事件响应

上一步找到最顶层的可响应视图后，从该视图的nextResponder开始，顺着这个链表遍历，直到该事件被处理掉，否则就会一直传递，直到丢弃。这个过程就是遍历单向链表的过程。

全局来看，nextResponder在运行时构成的数据结构是一个有向无环图，局部来看，从图中任意一节点开始，nextResponder指针构成的是一个单向链表，而事件响应就是从事件传递阶段找到的视图节点开始进行单向链表的遍历。

1. controller 的生命周期
2. layoutsubviews被调用的时机
3. drawrect 的调用时机
4. 布局管线 绘制管线 渲染管线，三层管线构成了iOS的布局刷新管线机制