**消息转发机制和响应链机制**

根据main函数的参数加载UIApplication－>AppDelegate->UIWindow->UIViewController->superView->subViews

关系为：UIApplication.keyWindow.rootViewController.view.subView

事件传递机制：

1.当iOS程序中发生触摸事件后，系统会将事件加入到UIApplication管理的一个任务队列中

2.UIApplication将处于任务队列最前端的事件向下分发。即UIWindow。

3.UIWindow将事件向下分发，即UIView。

4.UIView首先看自己是否能处理事件，触摸点是否在自己身上。如果能，那么继续寻找子视图。

5.遍历子控件，重复以上两步。

6.如果没有找到，那么自己就是事件处理者。如果

7.如果自己不能处理，那么不做任何处理。

其中 UIView不接受事件处理的情况主要有以下三种

1）alpha <0.01

2)userInteractionEnabled = NO

3.hidden ＝ YES

以下来自网络：

响应者链条概念: iOS系统检测到手指触摸(Touch)操作时会将其打包成一个UIEvent对象，并放入当前活动Application的事件队列，单例的UIApplication会从事件队列中取出触摸事件并传递给单例的UIWindow来处理，UIWindow对象首先会使用hitTest:withEvent:方法寻找此次Touch操作初始点所在的视图(View)，即需要将触摸事件传递给其处理的视图，这个过程称之为hit-test view。

UIResponder 是所有响应对象的基类，在UIResponder类中定义了处理上述各种事件的接口。我们熟悉的 UIApplication、 UIViewController、 UIWindow 和所有继承自UIView的UIKit类都直接或间接的继承自UIResponder，所以它们的实例都是可以构成响应者链的响应者对象。

UIWindow实例对象会首先在它的内容视图上调用hitTest:withEvent:，此方法会在其视图层级结构中的每个视图上调用pointInside:withEvent:（该方法用来判断点击事件发生的位置是否处于当前视图范围内，以确定用户是不是点击了当前视图），如果pointInside:withEvent:返回YES，则继续逐级调用，直到找到touch操作发生的位置，这个视图也就是要找的hit-test view。

hitTest:withEvent:方法的处理流程如下:

首先调用当前视图的pointInside:withEvent:方法判断触摸点是否在当前视图内；

若返回NO,则hitTest:withEvent:返回nil;

若返回YES,则向当前视图的所有子视图(subviews)发送hitTest:withEvent:消息，所有子视图的遍历顺序是从最顶层视图一直到到最底层视图，即从subviews数组的末尾向前遍历，直到有子视图返回非空对象或者全部子视图遍历完毕；

若第一次有子视图返回非空对象，则hitTest:withEvent:方法返回此对象，处理结束；

如所有子视图都返回非，则hitTest:withEvent:方法返回自身(self)。

一次完整的触摸事件的传递响应的过程

UIApplication --> UIWindow --> 递归找到最适合处理事件的控件

控件调用touches方法 --> 判断是否实现touches方法 --> 没有实现默认会将事件传递给上一个响应者 --> 找到上一个响应者

PS：如果直到UIApplication都不响应，那么这个事件就被废弃了。

1.响应者链条：由很多响应者链接在一起组合起来的一个链条

响应者：继承自UIResponder的对象称之为响应者对象

2.上一个响应者（默认做法是将事件顺着响应者链条向上传递，将事件交给上一个响应者进行处理）

如何判断当前响应者的上一个响应者是谁？

1>判断当前是否是控制器的View，如果是，上一个响应者就是控制器

2>如果当前不是控制器的View，上一个响应者就是父控件

3.响应者链条有什么用？

可以让一个触摸事件发声的时候让多个响应者同时响应该事件

在子类的实现文件里的touchesBegan：方法里加上如下代码即可

[super touchesBegan：touches withEvent：event]