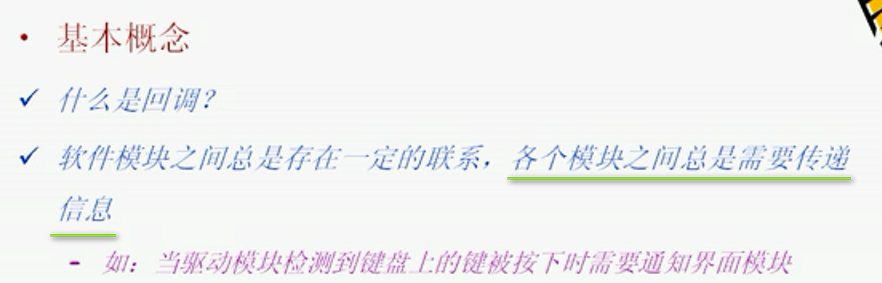


什么是回调？

说到回调 就要关注软件模块之间的联系 ---- 联系就是数据流动

同一个软件之间 各个模块是有联系的 模块之间如何联系？----- 就是数据流动



A模块的信息传递到B模块 --- 如何做到？

嵌入式里面 **通过消息队列的方式** ---- 局限在于 A B两个模块 不同的thread ---- A线程通知B线程 可以通过消息队列 实现 --- 比如A发送一个消息到消息队列 然后B线程就取得这个消息 通过这种方式

另外 A B不一定是线程 可以通过**函数指针** 这种方式更加灵活 这时候 A B两个模块不一定是线程了

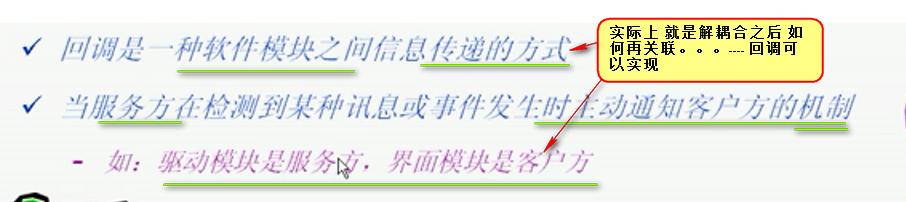
嵌入式软件最起码分成两部分：**驱动层和界面**

比如 驱动层检测到键盘按下一键 ----- 我们的记事本就显示你按下的字符

当驱动层检测到这个事件之后 ---- **产生了一个中断 硬件中断 这时候调用中断处理函数 isr** ---- 中断处理函数 就是调用一下这个函数 ---- 这时候可以传递一个函数指针 --- 函数指针就是界面模块的一个函数 ---- 这样 界面模块就收到了按下的按键 ---- 这时候界面模块就在屏幕上显示对应按下的按键

这样 两个模块分工明确 ----- 驱动模块检测到了按下按键之后 不会管检测到了 该做什么事情 只知道按下之后 我就通知你 ----- 给你的函数指针 ------ 至于你的函数指针做什么事情 我并不关心 ----- 只要通知就好

其实 回调实际上软件模块信息传递的方式



【这个就是我和四星说的 补全流程的一部分 回调函数实现】

比如上课着火了 打119进行灭火 但是 我们不关心怎么灭火 --- 是水枪喷 还是 二氧化碳进行灭火？ --- 这个打电话的号码 就是类比于一个函数的地址 只要知道这个函数的地址 怎么处理并不关心

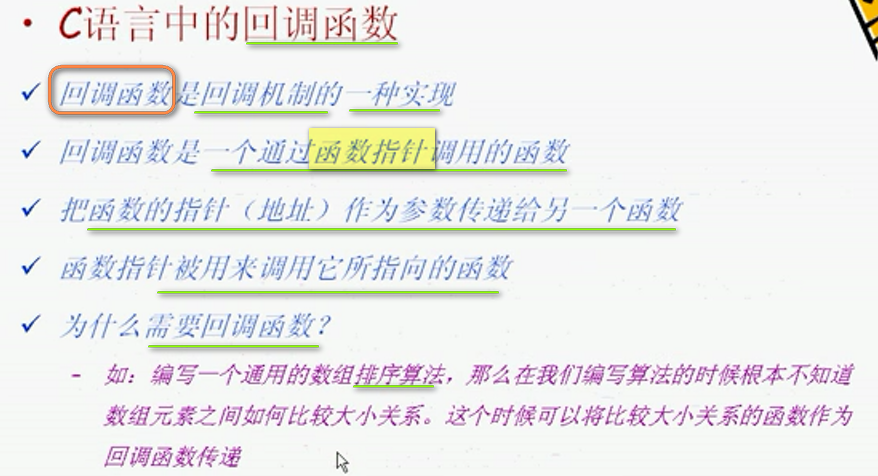
回调：

**事情发生的时候 去通知对方 ---- 对方到底干什么事 那是对方的事**

【这个就是回调和事件的关系】

**这样 耦合性非常弱 仅仅是指针**

看看C语言中回调函数



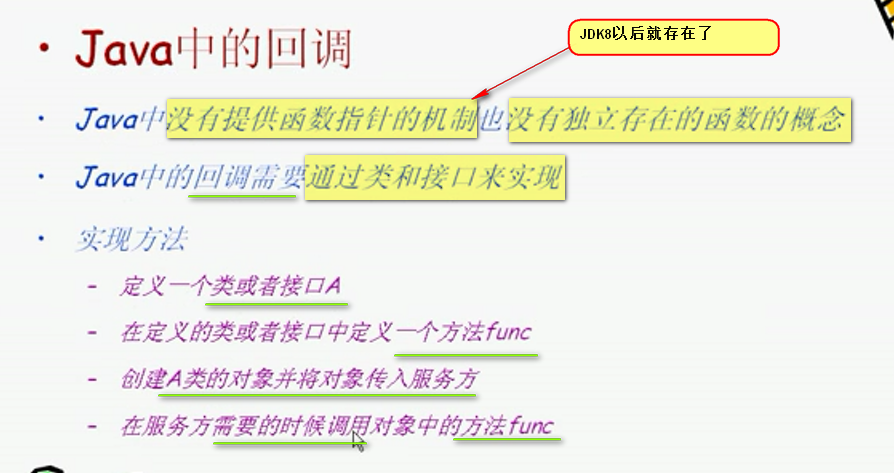
C语言中 是通过回调函数来实现回调的

为什么需要回调函数 --- 编写通用的排序算法 –牵扯到比较大小

之前排序都是整形数字 大小很好比较

但是 问题是 编写一个通用的排序算法 对String进行排序 这时候 如何比较大小？ 要传一个数组 和 比较大小的函数进去

这样 C语言编写通用的排序算法 不可避免遇到函数的指针



因此 java中没有函数指针

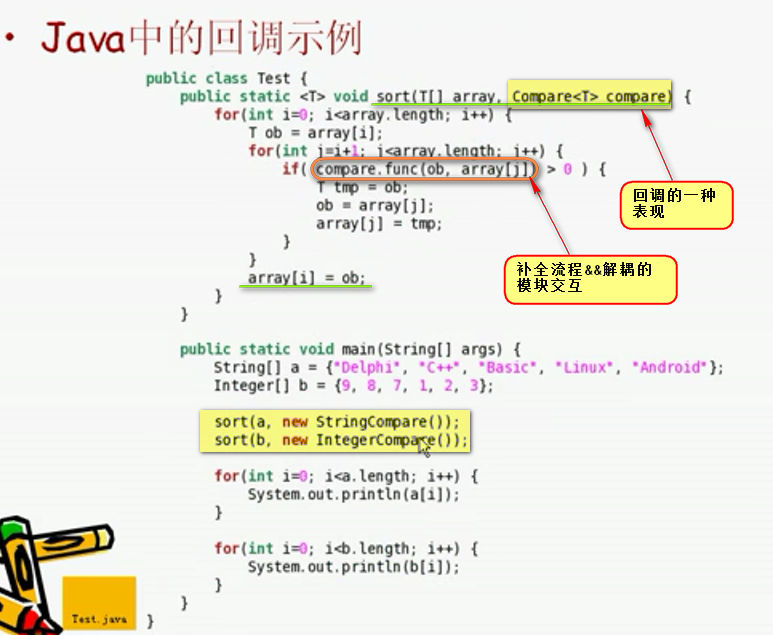
Java中的回调 是通过类和接口实现的

转了一个弯实现的回调

【所以 回调 就是 解耦之后 两个模块还要通信的一种手段 ---- 最终完成流程的补全】



这个a.compareTo(b)相当于c中的cmp方法



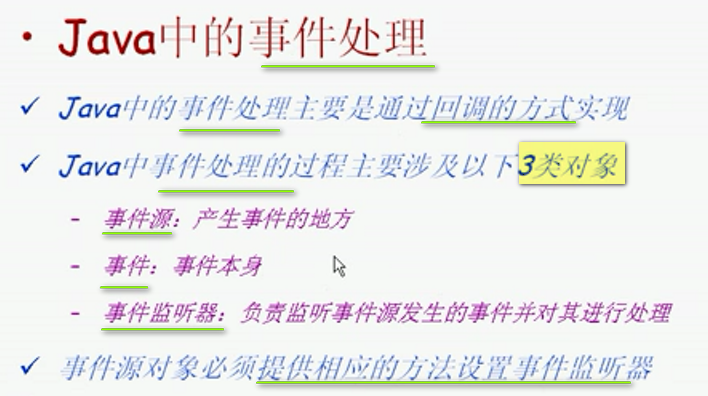
 这样写一个函数 各种类型的对象数组都能调用他进行排序

如果把这个例子放回C语言

Sort方法第二个参数是函数指针 ---- 遵循指针类型 是一致的才行 ------- Java变成了某种接口的类型

以上是java中的回调

============== Java中的事件处理 =================



刚才的例子 键盘就是事件源

事件本身 ---- 按下某个按键

回车键 ---- 记事本对回车键的处理不一样 按下去 是换行 ----- 事件处理过程中 一定有时间本身

**事件监听器实际上就是回调函数**

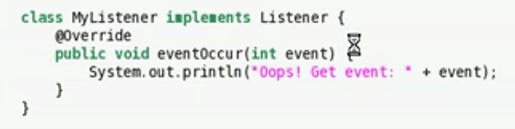
Java中的事件处理 助记：where(过去) + what +how(将来)

【三者之间如何联系？事件源本来和事件监听器构成一个完整的流程 ---- **也就是发生某一件事 然后如何处理这件事** --- 现在解耦合 变成了事件源 和 事件监听器 分别管理两部分 ---- 两部分如何传递相关公用的信息？把这个信息包装成事件 event ----- 然后 在事件源中 想办法把流程补全 所以 把数据包装成event 回调给事件监听器 ===== **所以 真正的流程在回调函数真正被调用的地方可以看到**

----- 观察者模式】

Java中的事件处理是回调方式实现 所以要先定义接口



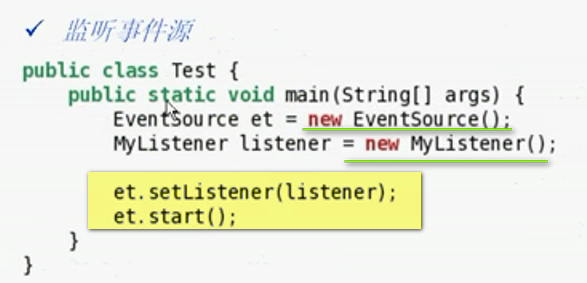


这里面event抽象成了int类型

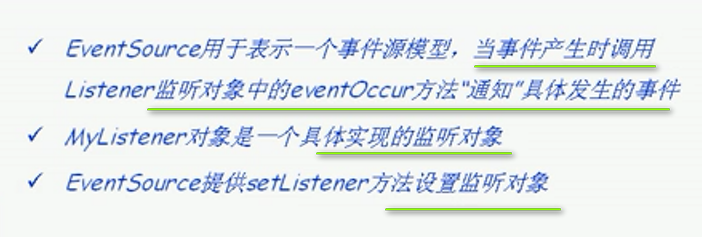


事件源 监听器的模型

用法

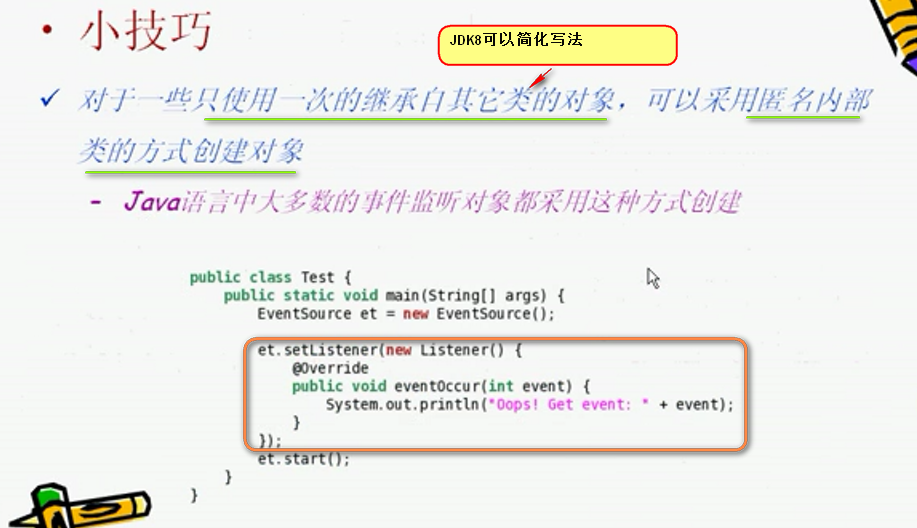






任何一个事件源 要提供一个事件监听器的摄入点

【事件如何产生 EventSource不管 只是负责进行产生事件 Listener利于分工的合作 --- 说白了 就是解耦合 驱动层 和 UI】



===============

