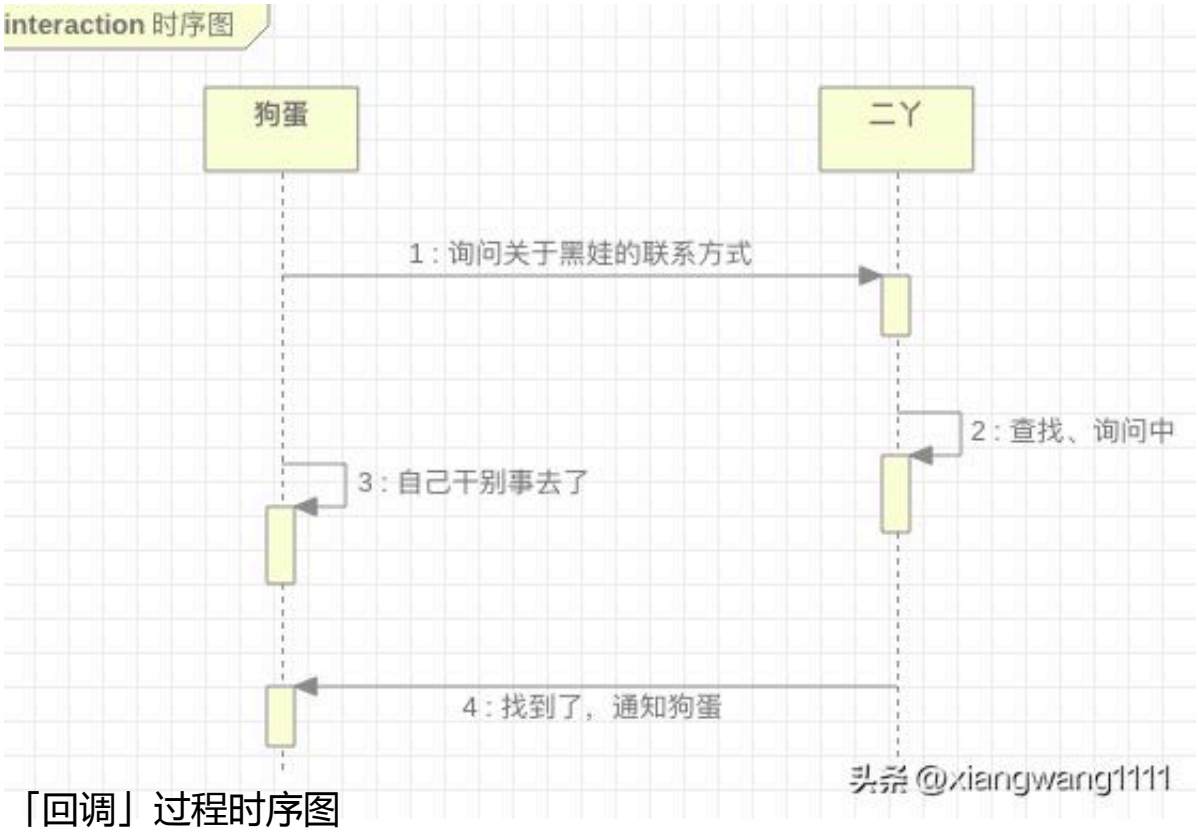
之前说了控制流中的四种，分别是顺序、分支、循环和递归（递归使用的场合较少，但却是无法替代的），现在就剩下最后一种了，这就是回调。

回调（念diào，不念tiáo）是编程专用术语，这个在其他领域都是没有的。回调的字面意思直接翻译过来就是：「回过头来调用」。

前面也说过，计算机干活有两种模式：同步和异步，拿狗蛋找黑娃的事例举了例子，这里继续拿他俩举例。狗蛋去找二丫问黑娃的电话，二丫说我要找找。但是这时候狗蛋没有「干等」，也没并有自己去问其他人，因为他妈妈打电话给他说有急事要他赶紧回家，临走时告诉二丫说：如果你找到了黑娃的电话，就给打电话告诉我，并且把自己的电话留给她了。

如果整个过程按照时间顺序展开，应该是有这么四个步骤：



「回调」过程时序图

这种做事的方式，在计算机编程里面，就叫做「回调」（可以理解为二丫回头调用狗蛋的联系方式通知他查找结果）。

更为通俗的例子是做家务：例如洗衣服。

- 女主人把衣服放到洗衣机里
- 洗衣机洗衣服，女主人去忙其他事情
- 洗衣机洗完衣服，发出通知（滴滴声）
- 女主人听到声音，来取出衣服晾晒

实现代码：

```
/**
 * ??????
 *
 * @author ?????
 */
```

```
public interface HouseWork {
    public void dry();
}

/**
 * ???
 *
 * @author ?????
 */
public class WashingMachine {
    private Vector<HouseWork> vector = new Vector<>();

    public void register(HouseWork work) {
        vector.add(work);
    }

    public void unregister(HouseWork work) {
        vector.remove(work);
    }

    public void notifyObserver() {
        for (HouseWork work : vector) {
            work.dry();
        }
    }

    public static void main(String[] args) throws Interrupte
dException {
        // ???
        WashingMachine machine = new WashingMachine();
        // ???
        Woman woman = new Woman();
        // ??????????????
        machine.register(woman);
        System.out.println("??????????");
        System.out.println("?????????");
        Thread.sleep(3000);
        // ???????????????
        machine.notifyObserver();
```

```
    }
}

/**
 * ???????????
 *
 * @author ?????
 */
public class Woman implements HouseWork {
    @Override
    public void dry() {
        System.out.println("??????");
    }
}
```

或者，使用更精简的代码来实现这个过程（需安装Java8以上版本）：

```
@FunctionalInterface
interface Wash {
    // ???
    public void finish();
}

/**
 * ???
 *
 * @author ?????
 */
public class Hostess {
    public void washClothes(Wash wash) {
        // ????????
        try {
            // ???
            TimeUnit.MILLISECONDS.sleep(3000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        // ?????
        wash.finish();
```

```
    }

    public void dosomething() {
        System.out.println("????????/???.....");
    }

    public static void main(String[] args) {
        System.out.println("?????????????");
        // ??????????
        new Thread(
                // ?????
                () -> new Hostess().washClothes(
                        // ???
                        () -> System.out.println("??????....
...???????")
                )
        ).start();
        // ????????
        new Hostess().dosomething();
    }
}
```

这就是完整的回调过程和示例演示。

对于初次接触回调的人来说，可能会有点绕，不好理解。尤其是回调和异步（《计算机干活的两种方式》中提到的方式）有什么区别？稍微有些基础的人还会问：回调和观察者模式又有什么区别？简单两句话就可以说清楚了：

首先，回调本质上也是一种异步模式。但很明显，狗蛋需要得到结果，而且是从别人那里得到结果；异步关注的则是「别干着等」这件事，有没有结果，相对来说不重要；

其次，观察者模式
是一种思维方法，而回调是一种具体实现。在观察者模式中，可以有多个观察者，而回调函数中的「观察者」只有一个。

至于什么是观察者模式，以后会讲。