# [spring事件监听（eventListener)](https://www.cnblogs.com/itplay/p/10982072.html)

### ****原理：观察者模式**[#](https://www.cnblogs.com/itplay/p/10982072.html" \l "2664508130)**

spring的事件监听有三个部分组成，事件（ApplicationEvent)、监听器(ApplicationListener)和事件发布操作。

### **事件[#](https://www.cnblogs.com/itplay/p/10982072.html" \l "4234480322)**

事件类需要继承ApplicationEvent，代码如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | public class HelloEvent extends ApplicationEvent {        private String name;        public HelloEvent(Object source, String name) {          super(source);          this.name = name;      }        public String getName() {          return name;      }  } |

这里为了简单测试，所以写的很简单。

事件类是一种很简单的pojo，除了需要继承ApplicationEvent也没什么了，这个类有一个构造方法需要super。

### **事件监听器[#](https://www.cnblogs.com/itplay/p/10982072.html" \l "3168843719)**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | @Component  public class HelloEventListener implements ApplicationListener<HelloEvent> {        private static final Logger logger = LoggerFactory.getLogger(HelloEventListener.class);        @Override      public void onApplicationEvent(HelloEvent event) {          logger.info("receive {} say hello!",event.getName());      }  } |

事件监听器需要实现ApplicationListener接口，这是个泛型接口，泛型类类型就是事件类型，其次需要是spring容器托管的bean，所以这里加了@component，只有一个方法，就是onApplicationEvent。

### **事件发布操作[#](https://www.cnblogs.com/itplay/p/10982072.html" \l "1234741828)**

有了事件和监听器，不发布事件也不用，事件发布方式很简单

|  |  |
| --- | --- |
| 1 | applicationContext.publishEvent(new HelloEvent(this,"lgb")); |

然后调用方法就能看到

|  |  |
| --- | --- |
| 1 | 2018-10-02 19:08:00.052  INFO 284928 --- [nio-5577-exec-3] l.b.e.c.s.event.HelloEventListener       : receive lgb say hello! |

### **疑问[#](https://www.cnblogs.com/itplay/p/10982072.html" \l "3495634292)**

1. 同样的事件能有多个监听器 -- 经过测试是可以的
2. 事件监听器一定要写一个类去实现吗 -- 其实是可以不需要的，spring有个注解@EventListener，修饰在方法上，稍后给出使用方法
3. 事件监听操作和发布事件的操作是同步的吗？ -- 是的，所以如果有事务，监听操作也在事务内

### **使用@EventListener注解[#](https://www.cnblogs.com/itplay/p/10982072.html" \l "3197203729)**

先给出代码

|  |  |  |
| --- | --- | --- |
| 1  2  3  4 | @EventListener  public void listenHello(HelloEvent event) {      logger.info("listen {} say hello from listenHello method!!!",event.getName());  } |  |

EventListener这个注解其实可以接受参数来表示事件源的，有两个参数classes和condition，顾明思议前者是表示哪一个事件类，后者是当满足什么条件是会调用该方法，但其实都可以不用用到，直接在方法上写参数HelloEvent就行