

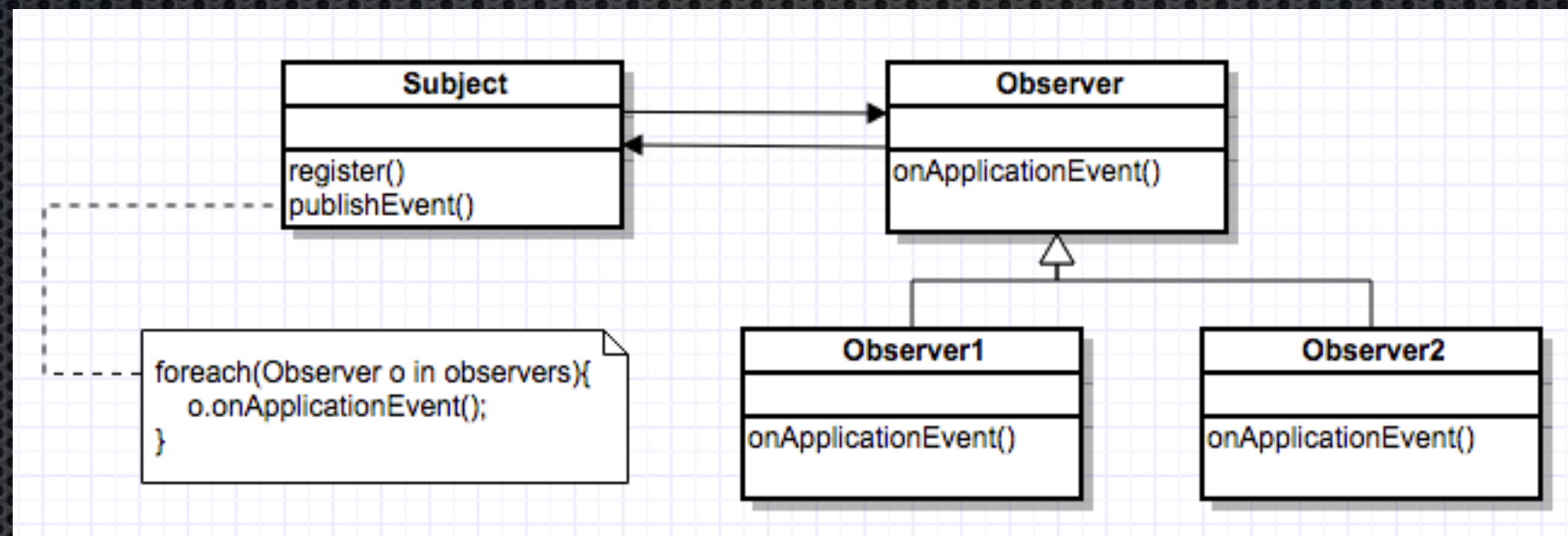
# Spring lesson 4

xieqiaoyun



# Event

Event是框架里面组件之间进行消息通讯的重要功能，通常使用观察者模式实现，剥离发送者和接受者





# 如何使用Spring的Event?

- ✦ Event对象继承自ApplicationEvent
- ✦ 发送者须注入ApplicationEventPublisher对象
- ✦ 接收者继承ApplicationListener



# Spring的Event

```
public static void main(String[] args) {  
    ApplicationContext ctx = new ClassPathXmlApplicationContext(  
        "applicationContext.xml");  
    LdOrderService sender=ctx.getBean(LdOrderService.class);  
    sender.createOrder();  
}
```

ctx注入一个实现了  
ApplicationEventPublisher接口的bean

```
@Component  
public class LdOrderService {  
    @Autowired  
    private ApplicationEventPublisher publisher;  
}
```

调用publishEvent, 执行multicastEvent方法

```
public void multicastEvent(final ApplicationEvent event, ResolvableType eventType) {  
    ResolvableType type = (eventType != null ? eventType : resolveDefaultEventType(event));  
    for (final ApplicationListener<?> listener : getApplicationListeners(event, type)) {  
        Executor executor = getTaskExecutor();  
        if (executor != null) {  
            executor.execute(() -> { invokeListener(listener, event); });  
        }  
        else {  
            invokeListener(listener, event);  
        }  
    }  
}  
protected void invokeListener(ApplicationListener listener, ApplicationEvent event) {  
    ErrorHandler errorHandler = getErrorHandler();  
    if (errorHandler != null) {  
        try {  
            listener.onApplicationEvent(event);  
        }  
        catch (Throwable err) {  
            errorHandler.handleError(err);  
        }  
    }  
    else {  
        try {  
            listener.onApplicationEvent(event);  
        }  
    }  
}
```



# Spring的Event

```
@Override  
public void onApplicationEvent(ApplicationEvent event) {  
    processEvent(event);  
}
```

↓ 容器根据@EventListener注册的Observer

```
public void processEvent(ApplicationEvent event) {  
    Object[] args = resolveArguments(event);  
    if (shouldHandle(event, args)) {  
        Object result = doInvoke(args);  
        if (result != null) {  
            handleResult(result);  
        }  
    }  
    else {  
        logger.trace("No result object given - no result to handle");  
    }  
}  
}
```

```
send order1: LDOrder{userId=1,price=200,createDate=Mon Sep 04 16:14:12 CST 2017}  
send order2: LDOrder{userId=2,price=600,createDate=Mon Sep 04 16:14:12 CST 2017}  
VIPUserService receive order: LDOrder{userId=2,price=600,createDate=Mon Sep 04 16:14:12 CST 2017}  
create vip user:VIPUser{userId=2,userName=VIP2,regDate=Mon Sep 04 16:14:12 CST 2017}  
you have upgraded to VIP user.  
send mail to: VIPUser{userId=2,userName=VIP2,regDate=Mon Sep 04 16:14:12 CST 2017}  
  
Process finished with exit code 0
```



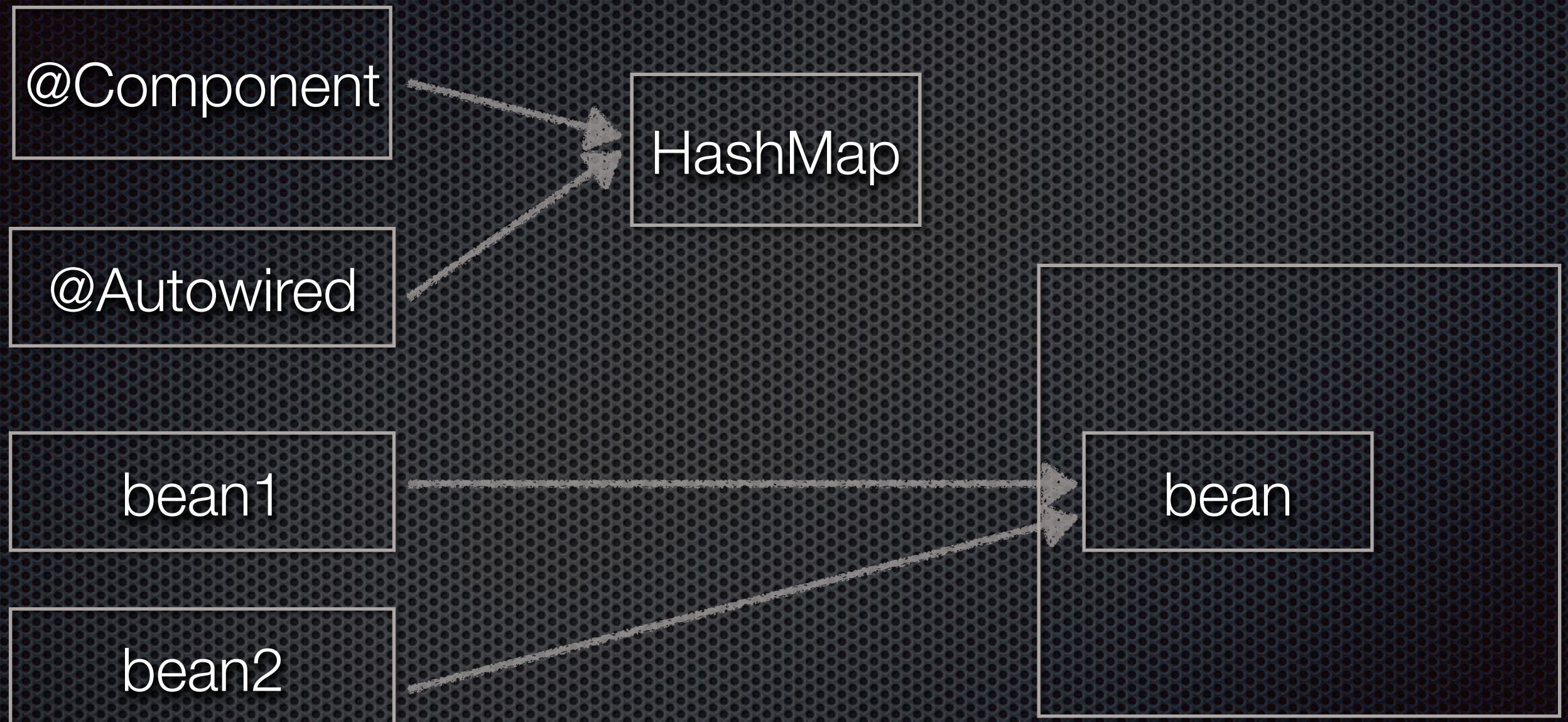
# Java注解&范型注入DIY IOC

我需要解决以下问题

- ✦ 如何扫描class
- ✦ 如何管理对象
- ✦ 如何解释annotation



# Java注解&范型注入DIY IOC





# Java注解&范型注入DIY IOC

范型注入还没写完。。。。





# Thank You

- 上次课周末写了，对模块类型的项目不会管理，还有点小尾巴需要处理