WAY2LEARN

LAB- Using Subscribable Channels in EI

In this lab you will be understanding how to use various Subscribable channels in spring integration

In this lab, you will be working on **03-using-subscribablechannels-start** project

Step1

In this step, you will understand how to use publish subscribable channels

1) Open 03-subscribable-channels-demo.xml and observe the configuration

Now, modify the configuration of orderschannel to make it as as publish subscribable channel as shown below :

```
      <int:publish-subscribe-channel id="orderschannel" />
```

Open SubscribableChannelsDemo.java and run it.

Why is the exception thrown?

Fix the problem by removing the poller from service activator and run again.
Now you should see that it works.

Can you tell which Thread is executing the service activator?

Yes. It is caller thread(in this case main thread).

2) Now configure one more service activator as shown below :

```
<int:service-activator input-channel="orderschannel"
                                   ref="myprocessor" method="doProcess"  />
```

For this service activator also, the input channel is same.

So, orderschannel has 2 subscribers.

Now Run SubscribableChannelsDemo and observe that both subscribers are invoked for each message.

Now change the configuration such that service activator for myprocessor bean is the first subscriber and run SubscribableChannelsDemo. (Hint : User order attribute of service-activator)

You should observe that myprocessor is invoked before orderprocessor

3) When you are using publish-subscribable channel, the messages are processed synchronously till now. i.e, sender thread is executing all the subscribers

Now configure the publish-subscribable channel to be asynchronous by attaching a task executor to it as shown below

Step 2

In this step, you will understand how to use Direct channels

1) Open 04-direct-channel-demo.xml and observe the configuration. Observe that there are 2 service activators subscribed to orders channel.

Now make the orders channel as direct channel.

Run DirectChannelDemo and observe that service activators are executed in round robin fashion.

2) Now disable round robin as shown below :

```
<int:channel id="orderschannel"  >
            <int:dispatcher load-balancer="none"/>
      </int:channel>
```

Run DirectChannelDemo and observe that first subscriber is invoked always.

3) Open OrderProcessServiceImpl and complete TODO-1 by throwing an exception

Run DirectChannelDemo and observe that second subscriber is invoked incase first subscriber throws an exception

This is because by default failover=true.

Now Disable failover as shown below :

```
      <int:channel id="orderschannel"  >
            <int:dispatcher failover="false" load-balancer="none"/>
      </int:channel>
```

Run DirectChannelDemo and observe that exception is propagated to the caller with out executing the second subscriber

4) Make the orderschannel as executor channel as shown below :

```
<task:executor id="te" pool-size="3-5" queue-capacity="20"/>

      <int:channel id="orderschannel"  >
            <int:dispatcher  task-executor="te"/>
      </int:channel>
```

Run DirectChannelDemo and observe that the subscribers are executed in a different thread.

Congratulations !! You know how to use all Subscribable Channels