**RabbitMQ**

**Configurações gerais**

No application.properties/yml:

spring.rabbitmq.host = host.docker.internal  
spring.rabbitmq.port = 5672  
spring.rabbitmq.username = guest  
spring.rabbitmq.password = guest

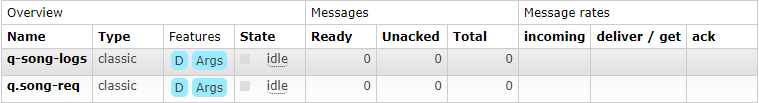
**Exchanges**

Uma **exchange** é a ferramenta que direciona as mensagens até as filas. Cada tipo de exchange tem as suas regras de direcionamento.

* **Fanout exchange**

Espalha a mensagem para todas as filas bindadas.

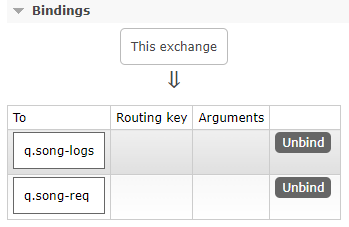
**Filas criadas:**



**Exchange criada:**

****

**Bindings:**

****

Código **producer** simples. A mensagem é enviada, em formato de json, para a exchange **x.song-req**, que será responsável por espalhar a mensagem às filas bindadas (**q.song-logs** e **q.song-req**):

@Service  
@AllArgsConstructor  
public class SongRequestService {  
  
 private RabbitTemplate rabbitTemplate;  
 private ObjectMapper objectMapper;  
  
 public SongRequest sendSongRequestToQueue(SongRequest songRequest) throws JsonProcessingException {  
 var json = objectMapper.writeValueAsString(songRequest);  
 rabbitTemplate.convertAndSend("x.song-req", "", json);  
 return songRequest;  
 }

Código **consumer** simples. Recebe a mensagem no formato json (String), e utiliza da biblioteca Gson para convertê-lo para uma instância do tipo SongRequest:

@Service  
@Slf4j  
@AllArgsConstructor  
public class SongPlayerService {  
  
 private Gson gson;  
  
 @RabbitListener(queues = "q.song-req")  
 public void listen(String msg){  
 SongRequest songRequest = gson.fromJson(msg, SongRequest.class);  
  
 playSong(songRequest);  
 }  
  
 private void playSong(SongRequest songRequest){  
 System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  
 System.*out*.println("Playing " + songRequest.getSong() + "...");  
 System.*out*.println("Requested by: " + songRequest.getRequestedBy());  
 System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  
 }  
}

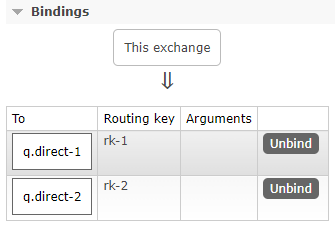
* **Direct exchange**

Direciona a mensagem para filas baseado no routing-key da mensagem.

Segue o mesmo código da **Fanout Exchange**, mas utiliza **routing-keys** para direcionar as mensagens, que seriam passadas no segundo argumento da função **“convertAndSend(..., ..., ...)**”.

Note que a função convertAndSend(..., ..., ...), não recebe uma routing-key no **Fanout Exchange**. Isso porque ela é designada para espalhar a mensagem para mais de uma fila.

As routing-keys são configuradas na exchange.



No exemplo acima, observamos que mensagens com a **routing-key** **rk-1** serão direcionadas para a fila **q.direct-1**, quanto as com a **rk-2** serão direcionadas para a fila **q.direct-2**.