GeeksHubs
academy _

Hello RabbitMQ

Experimentado con RabbitMQ

Fco. Marcos Palacios Rovira

@marcware82

GeeksHubs
an Alfatec group company

Slack alcoiacomtat-devs

Introducción

El objetivo de esta charla es mostrar y entender unos pequeño proyectos de ejemplo, sobre rabbitmq en PHP,

Se pueden encontrar en la documentación oficial y son un buen ejemplo práctico de cómo empezar a trabajar con colas.

https://geekshubs.slack.com/

Introducción RabbitMQ

Los ejemplos son de la documentación oficial:

Getting started with RabbitMQ

Todos los ejercicios en diferentes lenguajes
rabbitmq/rabbitmq-tutorials: Tutorials for using RabbitMQ in various ways

Ejercicios con PHP
https://github.com/rabbitmq/rabbitmq-tutorials/tree/master/php

Introducción RabbitMQ

¿Que es rabbitMQ?

- Es un sistema de mensajería multiprotocolo
  - Gestor de colas
  - AMQP es el protocolo por defecto
- Comunicación asíncrona

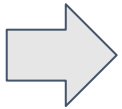
¿En que nos pueden ayudar?

- Reducir las cargas
- Tiempos de entregas por parte de los servidores
- Desacoplamiento entre servicios
- Escalabilidad

Actores en escena

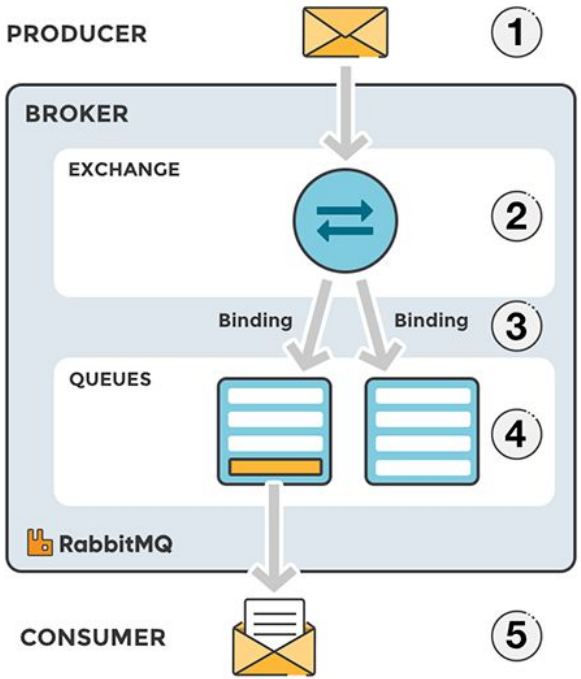
Producing **Exchange** **Queue** **Consuming**

Bindings



Routing keys

Flujo de envío



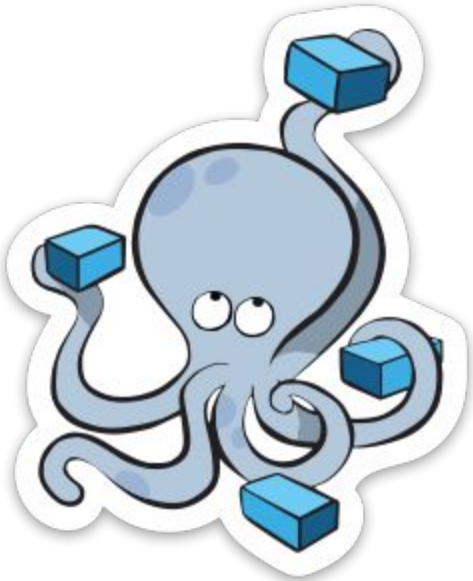
1. El **producer** publica los datos al **exchange**
2. El **exchange** recibe los datos y pasa a ser el responsable del enrutamiento.
3. Se debe establecer un **binding** entre la **Queues** y el **exchange**.
4. Los datos permanecen en la **queues** hasta que sean manejados por un **consumer**
5. El **consumer** procesa los datos

Docker Compose

Y con docker se ha montado la infraestructura.

Contenedores:

- php_sender
- php_receiver
- rabbitmq

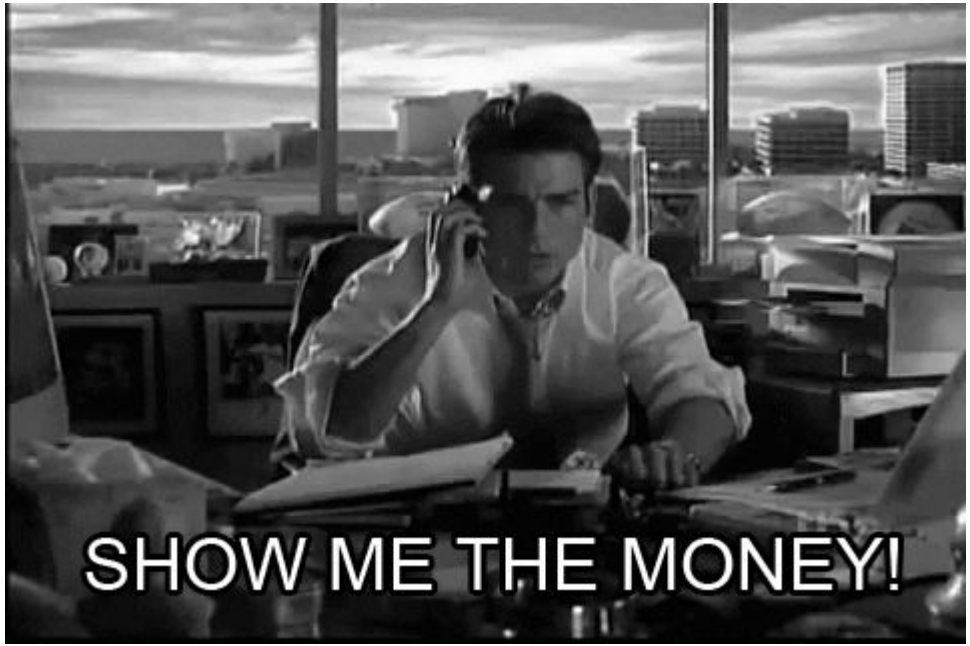


Composer

composer.json

```
{
  "require": {
    "php-amqplib/php-amqplib": ">=2.6.1"
  }
}
```

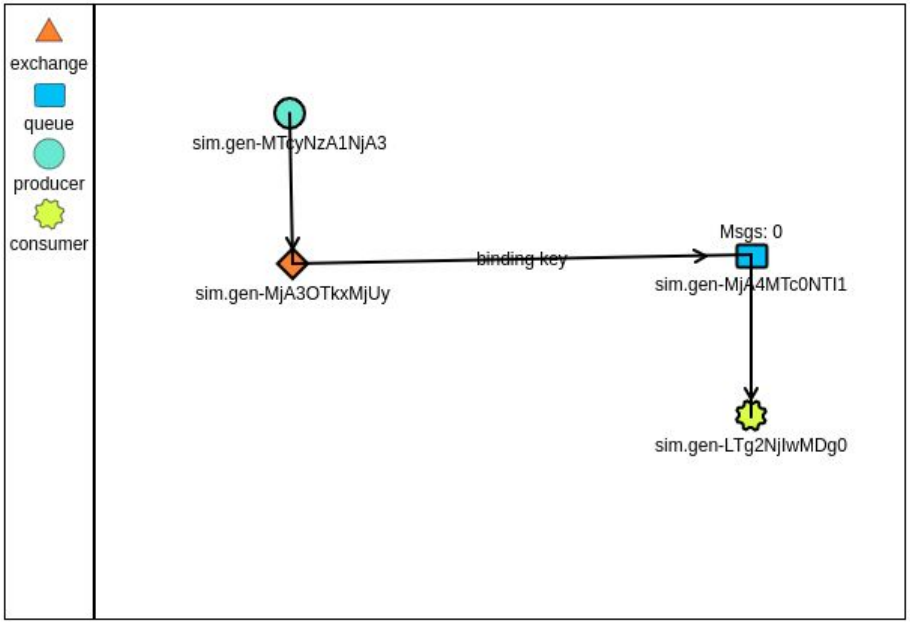
1 Hello World



1 Hello World

RabbitMQ Simulator

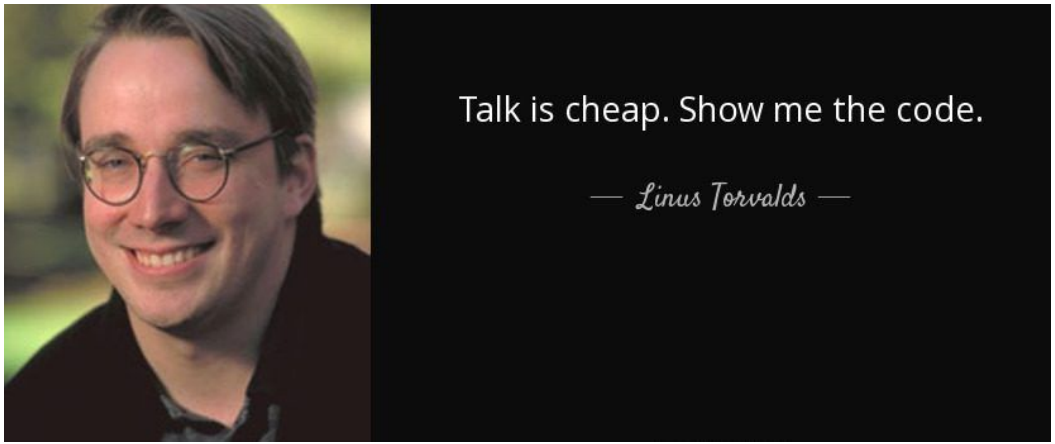
Local RabbitMQ queues



2 Work Queues

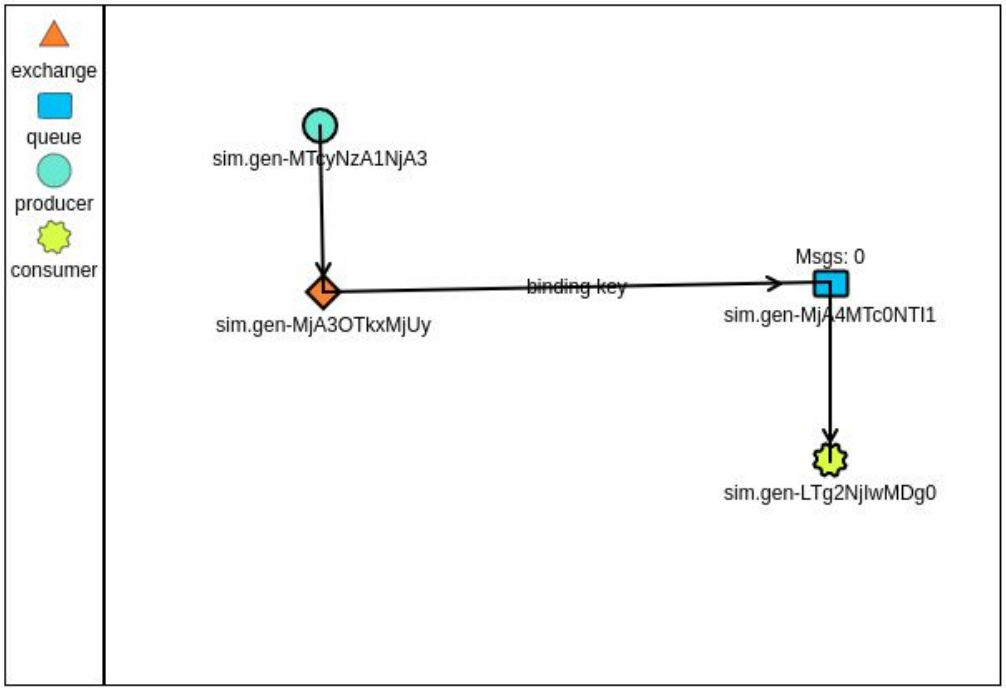
Mejora del Primer ejercicio:

- Los mensajes perduren si nadie los consume
- Que por cada Receive, consuma un mensaje a la vez
- Sleep por cada punto en el mensaje



2 Work Queues

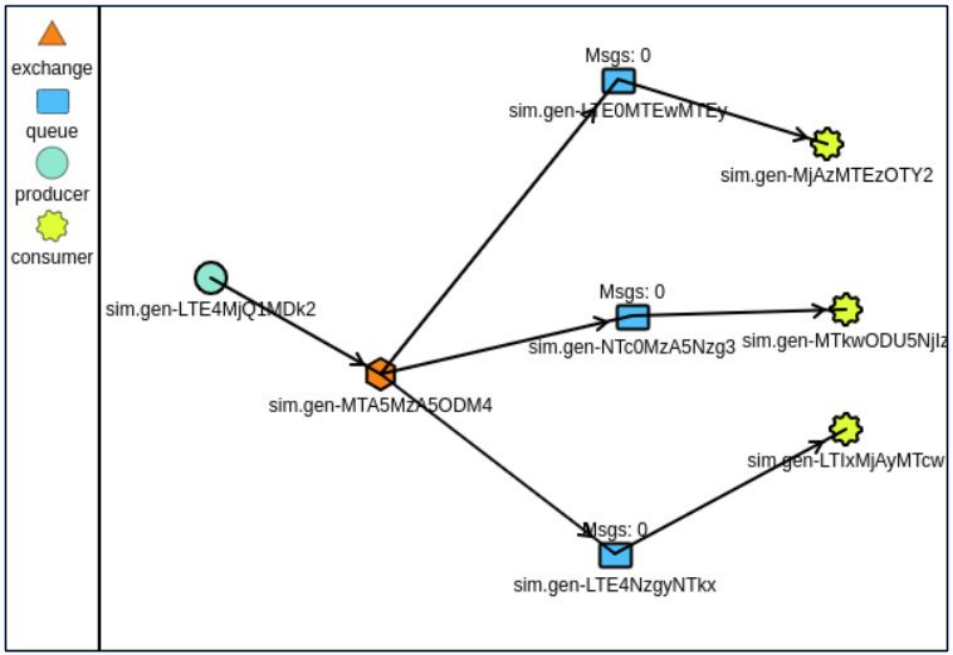
RabbitMQ Simulator

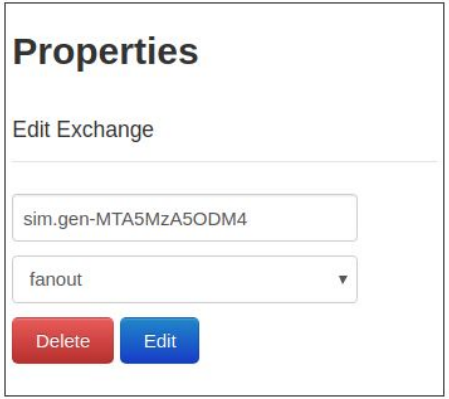


3 Publish Susbcribe

Tipo : fanout, es un broadcast

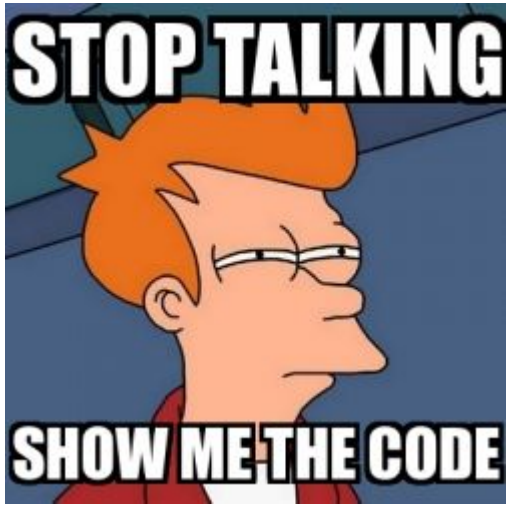
3 Publish Susbcribe



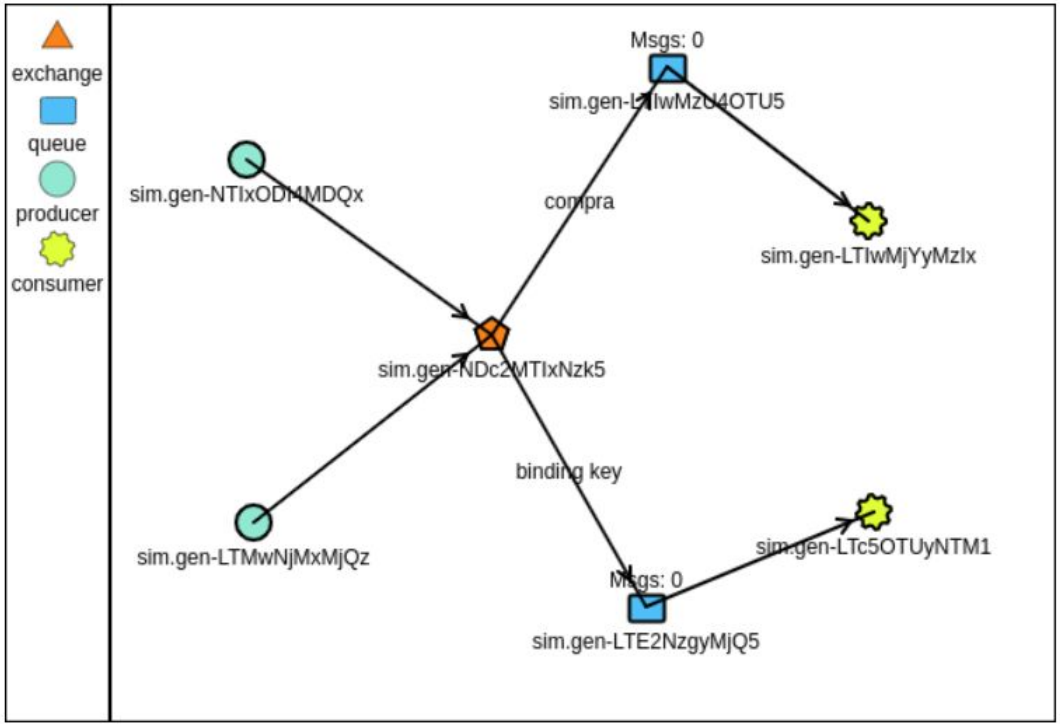


4 Routing

Introducimos Bindigs, mejorando el ejercicio de **Publish/Subscribe**

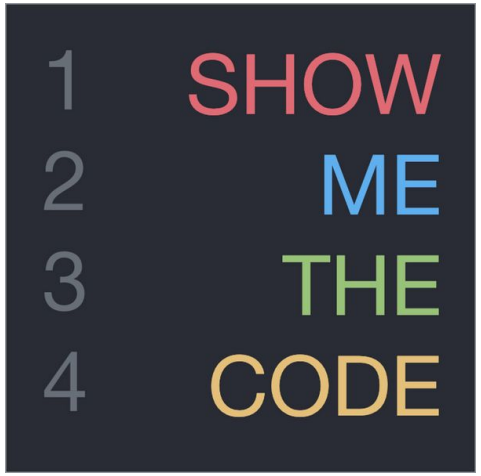


4 Routing


exchange
queue
producer
consumer
Msgs: 0
sim.gen-LTIwMzU4OTU5
sim.gen-NTIxODI4MDQx
compra
sim.gen-LTIwMjYyMzIx
sim.gen-NDc2MTIxNzk5
binding key
sim.gen-LTMwNjMxMjQz
sim.gen-LTc5OTUyNTM1
Msgs: 0
sim.gen-LTE2NzgyMjQ5

5 Topics

Continuando con el ejemplo anterior, ahora realizamos el exchange es tipo **Topic**



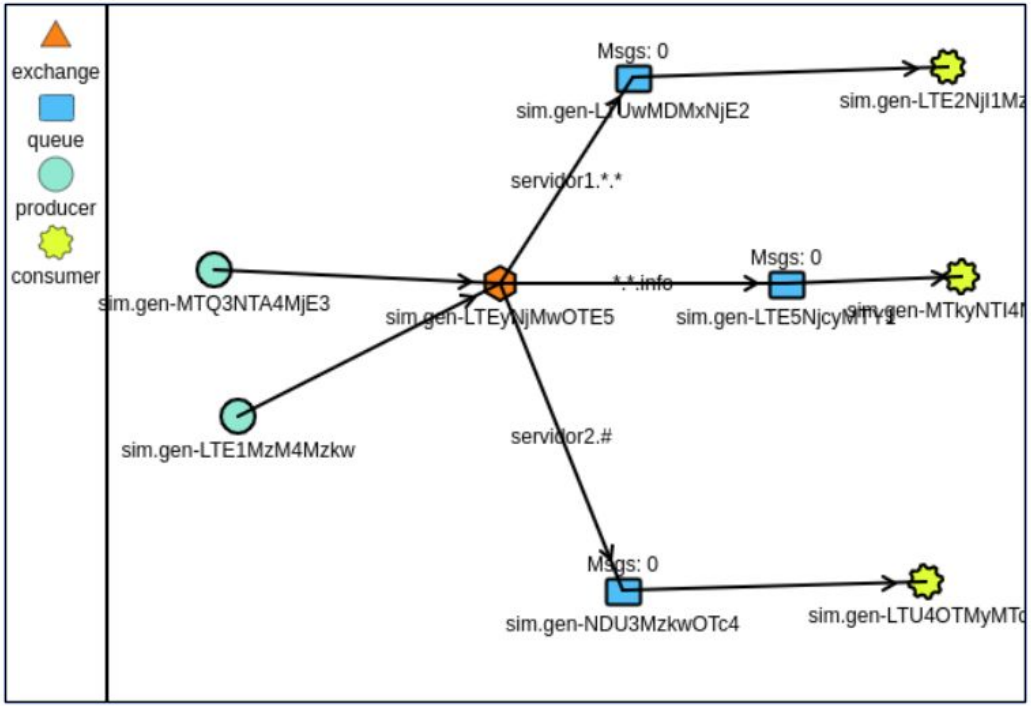
routing key, siguen patrones

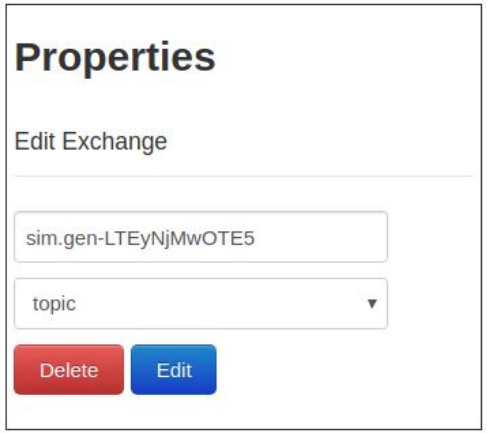
- Geekshubs.course.student

Binding Key

- Geekshubs.course.*
- Geekshubs.*.*
- Geekshubs.#
- *.course.*

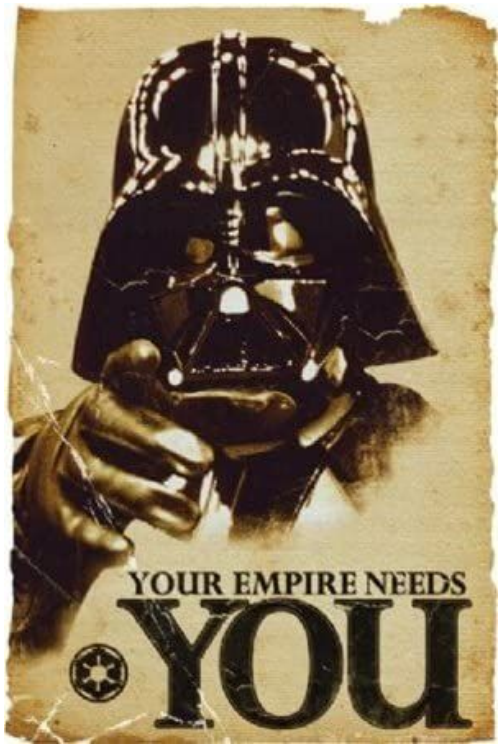
5 Topics


exchange
queue
producer
consumer
Msgs: 0
sim.gen-L
sim.gen-LTE2NjI1Mz
servidor1.*.*
sim.gen-MTQ3NTA4MjE3
sim.gen-LTEyNjMwOTE5
..info
Msgs: 0
sim.gen-LTE5NjcyMTY1
sim.gen-MTkyNTI4
sim.gen-LTE1MzM4Mzkw
servidor2.#
Msgs: 0
sim.gen-NDU3MzkwOTc4
sim.gen-LTU4OTMyMTc


Properties
Edit Exchange
sim.gen-LTEyNjMwOTE5
topic
Delete
Edit

6 RCP

¿Qué sucede si necesitamos ejecutar una función en una servidor remoto y esperar la respuesta?



Conclusiones

- Positivo
  - Arquitectura Bi-direccional sin bloqueos
  - Desacoplamiento
  - Escalabilidad
  - Comunicación indirecta
- Negativo
  - Complejidad
  - Conocimiento de la tecnología
  - Dificultad de testing

Bibliografía

Post de infraestructura
Getting started with PHP and RabbitMQ queues using Docker in style – Bizmate Solutions

Design Patterns: Competing Consumer Pattern
Patrón de consumidores de la competencia - Cloud Design Patterns

Vídeo de la comunidad PHPMad
Introducción a RabbitMQ por Alvaro Videla

Vídeo inspiración de la pruebas.
implementación del patrón competing consumers (work queues) mediante servidor RabbitMQ en la nube

¿?

GeeksHubs
an Alfatec group company

GeeksHubs
an Alfatec group company