

Artículo

[Daniel Aguilar](#) · 25 feb, 2022 Lectura de 6 min

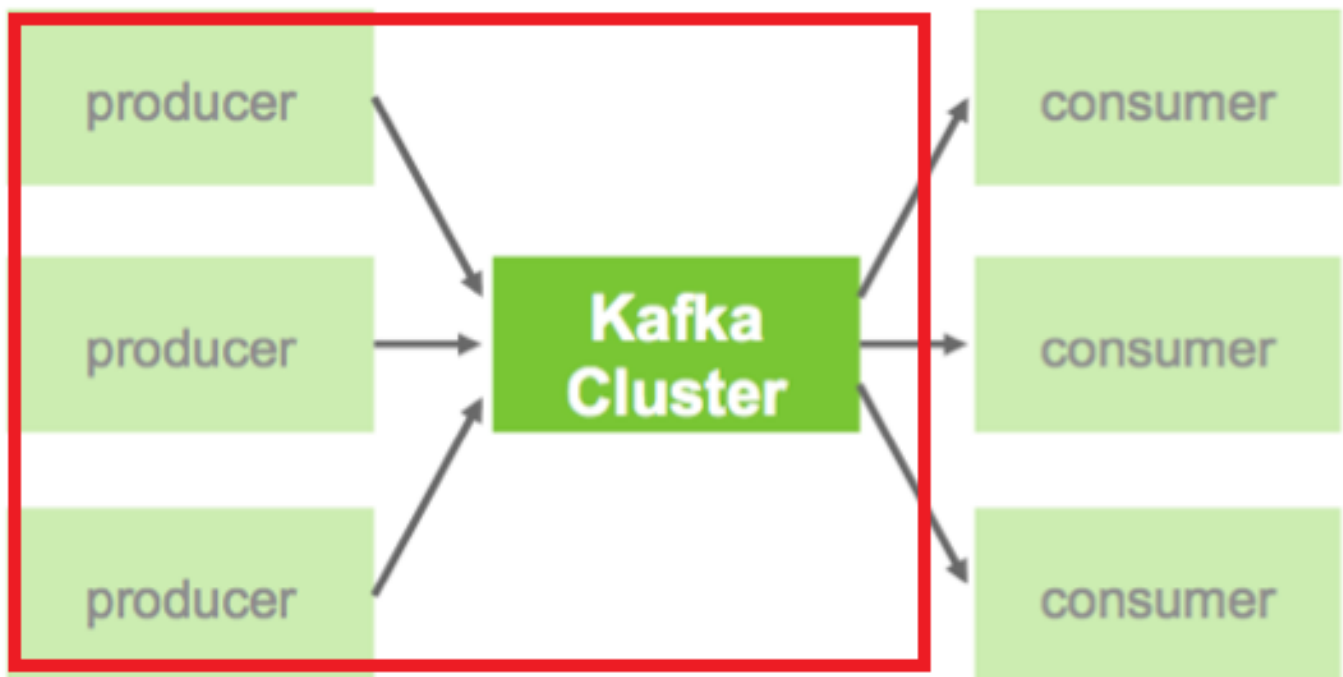
Enviando mensajes a Kafka

Hola!

Llevaba un tiempo queriendo dedicarle un rato para implementar alguna DLL o algo que pudiese usar desde Caché y al final he sacado un huequillo, si te interesa poder producir mensajes que se envíen a Kafka de una manera rápida estás en el lugar adecuado ;-)

Antes de daros la chapa con lo que vamos a ver os hago un resumen para que decidáis si os puede interesar leer el artículo.

En este artículo nos vamos a centrar "solo" en la parte de producir mensajes y enviarlos a Kafka:



¿Cómo funciona?

Uso una DLL de .Net (Netframework 4.5) que he hecho (está dentro de la carpeta dll del repositorio)

Después mediante 2 clases que he creado (puedes consultar el código fuente en el repositorio: [aqui](#)):

- Kafka.Helper.cls (Sirve para configurar e instalar todo).
- Kafka.Producer.cls (Sirve para crear mensajes y enviarlos a Kafka).

Podemos enviar mensajes a Kafka desde nuestras clases o rutinas.

Ejemplo:


```
SendObjectMessage()  
    N topic, message, res  
  
    Set topic="mitopic"  
  
    Set message=##class(%ZEN.proxyObject).%New()  
    Set message.nombre="Dani"  
    Set message.direccion="C/Falsa, 123"  
  
    Set res=##class(Kafka.Producer).sendObject(topic, .message)  
  
Q
```

¿Te interesa? Pues vamos a verlo en detalle:



La DLL en .Net usa una implementación de Confluent para enviar mensajes a un topic de Kafka (que debemos haber generado previamente, os dejo el fichero docker-compose que he creado para las pruebas, tendréis que modificar la dirección ip para hacerla coincidir con la ip del equipo donde lo despleguéis), (El fichero docker-compose es mejorable pero para las pruebas nos va a servir):

```
- /kafka/config:/kafka/config
environment:
  KAFKA_ADVERTISED_HOST_NAME: 172.16.172.10
  KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
kafka_manager:
  image: hlebalbau/kafka-manager:stable
```



Fichero docker-compose

Configuración en Caché y envío de mensaje:

Para configurarlo en Caché:

Copiaremos el contenido de la carpeta dll en una carpeta del servidor donde tenemos instalado Caché (yo lo que copiado dentro de c:\kafkaqs).

Después importaremos el paquete Kafka y abriremos la clase Kafka.Helper.cls

Dentro de la misma configuraremos los parámetros de configuración del Gateway que ejecuta la DLL

```
Class Kafka.Helper Extends %RegisteredObject
{

  Parameter port = 55000;

  Parameter host = "127.0.0.1";

  Parameter dll = "C:\kafkaqs\AQSKafkaDll7.dll";

  Parameter nombreGateway = "NetGateway";

  Parameter timeout = 2;
```

Si no sabes configurar el Gateway consulta este spoiler:

Configurar Gateway

Una vez configurado ejecutamos por terminal el metodo install:

```
Do ##class(Kafka.Helper).install()
```

A continuación procederemos a configurar el topic (Este topic debe estar definido previamente en Kafa)

Definir topic:

Creamos el topic en caché:

```
New topic,server,keyId
Set topic = "mitopic"
Set server = "172.16.172.5:9092"
Set keyId = "miKey" Do ##class(Kafka.Helper).createTopic(topic,server,keyId)
```

(En server debemos poner el puerto e ip configurados en el docker-compose)

Los topics creados se graban en el global ^KAFKA

Máscara búsqueda de global:

Historial de búsqueda:

Máximo

1:	<code>^KAFKA("TOPICS","mitopic")</code>	<code>= ""</code>
2:	<code>^KAFKA("TOPICS","mitopic","keyId")</code>	<code>= "miKey"</code>
3:	<code>^KAFKA("TOPICS","mitopic","server")</code>	<code>= "192.168.37.234:9092"</code>

Total: 3 [Fin de global]

La estructura del mismo es ^KAFA("TOPICS", nombreTopicX)

`^KAFA("TOPICS", nombreTopicX,"keyId")` = clave Id enviado junto los mensajes

`^KAFA("TOPICS", nombreTopicX,"server")` = ip kafka : puerto escucha

Podemos generar tantos topics como queramos.

Una vez tenemos todo configurado por fin podemos empezar a enviar mensajes!!

He implementado 2 metodos para hacerlo

1 - Envía un texto en formato JSON

2 - Envía un objeto %ZEN.proxyObject

3 - No he tenido tiempo pero si os gusta implementaré envío con dynamicObjects y objetos de clases definidas también.

Opción 1 Enviar texto en formato JSON:

```
Set topic="mitopic"
Set message="{ \"Valor\":\"hola\" }"
Set res=##class(Kafka.Producer).sendMessage(topic,message)
```

Opción 2 Enviar objeto %ZEN.proxyObject

```
Set topic="mitopic"

Set message=##class(%ZEN.proxyObject).%New()
Set message.nombre="Dani"
Set message.direccion="C/Falsa, 123"

Set res=##class(Kafka.Producer).sendObject(topic, .message)
```

Cuando enviamos un mensaje si por algún casual falla la comunicación o no puede enviar el mensaje por cualquier motivo deja registrado el error en un subnivel del global ^KAFKA dentro del topic que ha fallado:

```
If (resMessage'="enviado ok")
{
    Set sequence = $I(^KAFKA("TOPICS",topic,"ERRORS"))
    Set ^KAFKA("TOPICS",topic,"ERRORS",sequence,"MESSAGE")=message
    Set ^KAFKA("TOPICS",topic,"ERRORS",sequence,"RESULT")=resMessage
}
```

Lo que graba en el nivel "MESSAGE" es el mensaje en formato JSON (ya se haya pasado como objeto o como texto)

En resMessage se graba el resultado devuelto por la DLL de .Net.

Con esto ya podríamos enviar mensajes a Kafka, evidentemente habría que configurar algún subscriptor para los topics para que los mensajes que enviemos puedan ser procesados y ejecuten su función.

¿Que cuales podrían ser? Cualquiera!!, Enviar un e-mail, SMS, generar un PDF, sincronizar la información recibida con otro sistema... en fin cualquier cosa que se nos ocurra.

Personalmente me parece una herramienta muy muy potente y que nos permite desacoplar nuestros desarrollos y mantener las lógicas separadas.

Espero que os guste la idea, voy a seguir desarrollándola en mis ratos libres por lo que si tenéis cualquier sugerencia o mejora será mas que bienvenida.

Gracias por leer el artículo, y espero que os haya gustado.

Gracias por leerlo!!.

[#.NET](#) [#Concurso](#) [#ObjectScript](#) [#Caché](#)

URL de fuente: <https://es.community.intersystems.com/post/enviando-mensajes-kafka>