Artigo
Danusa Calixto · Set. 13, 2022 30min de leitura

Aproveitando a API de mensagens externas no InterSystems IRIS

A interoperabilidade é cada vez mais importante atualmente. O InterSystems IRIS 2022.1 tem uma nova API de mensagens para a comunicação com plataformas de streaming de eventos, como Kafka, AWS SQS/SNS, JMS e RabbitMQ.

Este artigo mostra como você pode se conectar ao Kafka e AWS SQS com facilidade.
Começamos com uma breve discussão dos conceitos e termos básicos das plataformas de streaming de eventos.

Finalidade das plataformas de streaming de eventos e termos comuns

As plataformas de streaming de eventos, como Kafka ou AWS SQS, são capazes de consumir um stream de eventos sem restrições em uma frequência bastante alta e podem reagir a eventos. Os consumidores leem os dados dos streams para processamento adicional. Eles são geralmente usados em ambientes de IoT.

Os termos comuns nessa área são:

- **Tópico/Fila**, o local onde os dados são armazenados
- **Produtor**, cria e envia os dados (eventos, mensagens) para um tópico ou uma fila
- **Consumidor**, lê os eventos/mensagens de um ou mais tópicos ou filas
- **Publicar/Assinar**, os produtores enviam os dados para um tópico/fila (publicar), os consumidores assinam um tópico/fila e recebem notificações automáticas com a chegada de novos dados
- **Polling**, os consumidores precisam buscar ativamente novos dados em um tópico/fila

Por que eles são usados?

- Dissociação de produtores e consumidores
- Altamente escalonável para dados em tempo real

Preciso mesmo deles? Como desenvolvedor do InterSystems IRIS, provavelmente não, mas você não está sozinho...

API de mensagens externas

As novas classes da API estão localizadas no pacote ***%External.Messaging***. Ele contém as classes genéricas ***Client-***, ***Settings-*** e ***Message***. As classes especializadas do Kafka, AWS SQS/SNS, JMS e RabbitMQ são subclasses dessas classes genéricas.

O fluxo de comunicação básico é:

1. Criar um objeto de configurações para a plataforma de destino. Isso também é responsável pela autenticação em relação à plataforma de destino.
2. Criar um objeto de cliente específico e transmitir o objeto de configurações a ele.
3. Criar um objeto de mensagem e enviar para o destino.

As seguintes seções demonstram como você pode se comunicar com Kafka e AWS SQS (Simple Queue Service).

Interação com o Kafka

Vamos começar com um exemplo do Kafka. Primeiro, criamos uma classe que usa a nova API %External Messaging para criar um tópico, enviar uma mensagem ao Kafka e receber uma mensagem dele.

Primeiro, ela cria um objeto de configurações do Kafka:

```
&lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-keyword"
>##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaSettings&l
t;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
&lt;span class="hljs-keyword">set&lt;/span> tSettings.servers = &lt;span class="hljs-
built_in">$$$KAFKASERVER&lt;/span>
&lt;span class="hljs-keyword">set&lt;/span> tSettings.groupId = &lt;span class="hljs-
string">"iris-consumer"&lt;/span>
```

Após definir o endereço do servidor do Kafka, ela define um ID de grupo do Kafka.

Com essas configurações, um objeto de cliente do Kafka é criado:

```
&lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-keyword">#
#class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt;/span>)
.CreateKafkaClient(tSettings.ToJSON(),.tSc)
```

Em seguida, um topico é criado ao invocar o método CreateTopic() do cliente do Kafka:

```
&lt;span class="hljs-keyword">Set&lt;/span> tSC = tClient.CreateTopic(pTopicName,tNum
berOfPartitions,tReplicationFactor)
```

Veja abaixo a amostra de código completa:

```
Include Kafka.Settings

&lt;span class="hljs-keyword">Class&lt;/span> Kafka.api [ Abstract ]
{

&lt;span class="hljs-keyword">ClassMethod&lt;/span> CreateTopic(pTopicName &lt;span c
lass="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>) &l
t;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaSe
ttings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.servers = &lt;span clas
s="hljs-built_in">$$$KAFKASERVER&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.groupId = &lt;span clas
s="hljs-string">"iris-consumer"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateKafkaClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
```

```
        &lt;span class="hljs-keyword">Set&lt;/span> tNumberOfPartitions = &lt;span cl
ass="hljs-number">1&lt;/span>
        &lt;span class="hljs-
keyword">Set&lt;/span> tReplicationFactor = &lt;span class="hljs-number">1&lt;/span>
        &lt;span class="hljs-keyword">Set&lt;/span> tSC = tClient.CreateTopic(pTopicN
ame,tNumberOfPartitions,tReplicationFactor)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSC)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tClient.&lt;span cla
ss="hljs-keyword">Close&lt;/span>())
        }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}



}
```

Depois de criar um tópico, podemos enviar e receber mensagens do Kafka. O código é semelhante ao exibido acima.

```
&lt;span class="hljs-keyword">ClassMethod&lt;/span> SendMessage(pMessage &lt;span cla
ss="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>, pTop
ic &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&
lt;/span>) &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tMessage &lt;span class="hljs-keywor
d">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaMessage&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaSe
ttings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.servers = &lt;span clas
s="hljs-built_in">$$$KAFKASERVER&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.groupId = &lt;span clas
s="hljs-string">"iris-consumer"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tMessage = &lt;span class="hljs-k
eyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaMes
sage&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tMessage.topic = pTopic
        &lt;span class="hljs-keyword">set&lt;/span> tMessage.value = pMessage
        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateKafkaClient(tSettings.ToJSON(),.tSc)
```

```
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-keyword">Set&lt;/span> producerSettings = &lt;span class
="hljs-string">"{""key.serializer"":""org.apache.kafka.common.serialization.StringSer
ializer""}"&lt;/span>
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.UpdateProducerConfig(producerSettings))
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.SendMessage(tMessage))
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tClient.&lt;span cla
ss="hljs-keyword">Close&lt;/span>())

    }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

&lt;span class="hljs-keyword">ClassMethod&lt;/span> ReceiveMessage(pTopicName &lt;spa
n class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>,
 pGroupId &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%
String&lt;/span> = &lt;span class="hljs-string">"iris-consumer"&lt;/span>, Output pMe
ssages) &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tMessage &lt;span class="hljs-keywor
d">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaMessage&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaSe
ttings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.servers = &lt;span clas
s="hljs-built_in">$$$KAFKASERVER&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.groupId = pGroupId

        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateKafkaClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-keyword">Set&lt;/span> producerSettings = &lt;span class
="hljs-string">"{""key.serializer"":""org.apache.kafka.common.serialization.StringSer
ializer""}"&lt;/span>
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.UpdateProducerConfig(producerSettings))
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.ReceiveMessage(pTopicName, .pMessages))
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tClient.&lt;span cla
ss="hljs-keyword">Close&lt;/span>())
```

```
        }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}
```

Vamos testar. Com uma instância do Kafka em execução, criamos um tópico ***community*** com o método ***CreateTopic*** acima:

```
EVENTS>set tSc = ##class(Kafka.api).CreateTopic("community")
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

EVENTS>write tSc
1
```

Ignore os avisos log4j aqui. O método retorna um código de status OK. Então, o tópico foi criado. Em seguida, vamos enviar uma mensagem para esse tópico. Para verificar que a mensagem foi enviada ao tópico, executo um consumidor do Kafka genérico. Esse consumidor ouve o tópico ***community***.

Então, vamos enviar uma mensagem para esse tópico. Vou enviar uma JSON-String para ele, mas você basicamente pode enviar qualquer formato de mensagem para um tópico.

Vamos conferir se o consumidor recebeu a mensagem:

A mensagem foi recebida com sucesso pelo consumidor.

Para receber mensagens e excluir tópicos, é parecido com a amostra acima. Veja abaixo a implementação da amostra completa. O arquivo include ***Kafka.settings*** só contém uma definição macro: ***#define KAFKASERVER <Kafka server location and port>***.

```
Include Kafka.Settings

&lt;span class="hljs-keyword">Class&lt;/span> Kafka.api [ Abstract ]
{

&lt;span class="hljs-keyword">ClassMethod&lt;/span> CreateTopic(pTopicName &lt;span c
lass="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>) &l
t;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaSe
ttings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.servers = &lt;span clas
s="hljs-built_in">$$$KAFKASERVER&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.groupId = &lt;span clas
s="hljs-string">"iris-consumer"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
```

```
;/span>).CreateKafkaClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-keyword">Set&lt;/span> tNumberOfPartitions = &lt;span cl
ass="hljs-number">1&lt;/span>
        &lt;span class="hljs-
keyword">Set&lt;/span> tReplicationFactor = &lt;span class="hljs-number">1&lt;/span>
        &lt;span class="hljs-keyword">Set&lt;/span> tSC = tClient.CreateTopic(pTopicN
ame,tNumberOfPartitions,tReplicationFactor)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSC)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tClient.&lt;span cla
ss="hljs-keyword">Close&lt;/span>())
        }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

&lt;span class="hljs-keyword">ClassMethod&lt;/span> DeleteTopic(pTopicName &lt;span c
lass="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>) &l
t;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaSe
ttings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.servers = &lt;span clas
s="hljs-built_in">$$$KAFKASERVER&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.groupId = &lt;span clas
s="hljs-string">"iris-consumer"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateKafkaClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-keyword">Set&lt;/span> tNumberOfPartitions = &lt;span cl
ass="hljs-number">1&lt;/span>
        &lt;span class="hljs-
keyword">Set&lt;/span> tReplicationFactor = &lt;span class="hljs-number">1&lt;/span>
        &lt;span class="hljs-
keyword">Set&lt;/span> tSC = tClient.DeleteTopic(pTopicName)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSC)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tClient.&lt;span cla
ss="hljs-keyword">Close&lt;/span>())
        }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

&lt;span class="hljs-keyword">ClassMethod&lt;/span> SendMessage(pMessage &lt;span cla
ss="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>, pTop
ic &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&
```

```
lt;/span>) &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tMessage &lt;span class="hljs-keywor
d">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaMessage&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaSe
ttings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.servers = &lt;span clas
s="hljs-built_in">$$$KAFKASERVER&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.groupId = &lt;span clas
s="hljs-string">"iris-consumer"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tMessage = &lt;span class="hljs-k
eyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaMes
sage&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tMessage.topic = pTopic
        &lt;span class="hljs-keyword">set&lt;/span> tMessage.value = pMessage
        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateKafkaClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-keyword">Set&lt;/span> producerSettings = &lt;span class
="hljs-string">"{""key.serializer"":""org.apache.kafka.common.serialization.StringSer
ializer""}"&lt;/span>
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.UpdateProducerConfig(producerSettings))
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.SendMessage(tMessage))
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tClient.&lt;span cla
ss="hljs-keyword">Close&lt;/span>())

    }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

&lt;span class="hljs-keyword">ClassMethod&lt;/span> ReceiveMessage(pTopicName &lt;spa
n class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>,
 pGroupId &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%
String&lt;/span> = &lt;span class="hljs-string">"iris-consumer"&lt;/span>, Output pMe
ssages) &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
```

```
built_in">%External.Messaging.KafkaSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tMessage &lt;span class="hljs-keywor
d">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.KafkaMessage&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.KafkaSe
ttings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.servers = &lt;span clas
s="hljs-built_in">$$$KAFKASERVER&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.groupId = pGroupId

        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateKafkaClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-keyword">Set&lt;/span> producerSettings = &lt;span class
="hljs-string">"{""key.serializer"":""org.apache.kafka.common.serialization.StringSer
ializer""}"&lt;/span>
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.UpdateProducerConfig(producerSettings))
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.ReceiveMessage(pTopicName, .pMessages))
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tClient.&lt;span cla
ss="hljs-keyword">Close&lt;/span>())
        }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

}
```

Interação com o AWS SQS

Como você se comunicaria com o AWS SQS (Simple Queue Service)?
O procedimento básico é bastante similar. No entanto, o AWS exige a autenticação e não usa o termo "tópico". Ele fala sobre filas. Você pode enviar uma mensagem para uma fila, e os consumidores podem receber mensagens de uma ou mais filas.

Semelhante à classe da API acima, criei algo para o AWS SQS.

```
&lt;span class="hljs-keyword">Class&lt;/span> AWS.SQS.api [ Abstract ]
{

&lt;span class="hljs-keyword">ClassMethod&lt;/span> SendMessage(pMessage &lt;span cla
ss="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>, pQue
```

```
ue &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&
lt;/span>) &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.SQSSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tMessage &lt;span class="hljs-keywor
d">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.SQSMessage&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
">as&lt;/span> &lt;span class="hljs-built_in">%External.Messaging.SQSClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(&lt;span class="hljs-
keyword">##class&lt;/span>(AWS.Utils).GetCredentials(.tCredentials))
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.SQSSett
ings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.accessKey = tCredential
s(&lt;span class="hljs-string">"aws_access_key_id"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.secretKey = tCredential
s(&lt;span class="hljs-string">"aws_secret_access_key"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.sessionToken = tCredent
ials(&lt;span class="hljs-string">"aws_session_token"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.region = &lt;span class
="hljs-string">"eu-central-1"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tMessage = &lt;span class="hljs-k
eyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.SQSMessa
ge&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tMessage.body = pMessage
        &lt;span class="hljs-keyword">set&lt;/span> tMessage.queue = pQueue

        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateSQSClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.SendMessage(tMessage))

    }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

&lt;span class="hljs-keyword">ClassMethod&lt;/span> ReceiveMessage(pQueueName &lt;spa
n class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>,
 Output pMessages) &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.SQSSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
```

```
">as&lt;/span> &lt;span class="hljs-built_in">%External.Messaging.SQSClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(&lt;span class="hljs-
keyword">##class&lt;/span>(AWS.Utils).GetCredentials(.tCredentials))
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.SQSSett
ings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.accessKey = tCredential
s(&lt;span class="hljs-string">"aws_access_key_id"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.secretKey = tCredential
s(&lt;span class="hljs-string">"aws_secret_access_key"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.sessionToken = tCredent
ials(&lt;span class="hljs-string">"aws_session_token"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.region = &lt;span class
="hljs-string">"eu-central-1"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateSQSClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.ReceiveMessage(pQueueName, .pMessages))

    }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

&lt;span class="hljs-keyword">ClassMethod&lt;/span> DeleteMessage(pQueueName &lt;span
 class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>,
pReceiptHandle &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_
in">%String&lt;/span>) &lt;span class="hljs-
keyword">As&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.SQSSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
">as&lt;/span> &lt;span class="hljs-built_in">%External.Messaging.SQSClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(&lt;span class="hljs-
keyword">##class&lt;/span>(AWS.Utils).GetCredentials(.tCredentials))
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.SQSSett
ings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.accessKey = tCredential
s(&lt;span class="hljs-string">"aws_access_key_id"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.secretKey = tCredential
s(&lt;span class="hljs-string">"aws_secret_access_key"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.sessionToken = tCredent
ials(&lt;span class="hljs-string">"aws_session_token"&lt;/span>)
```

```
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.region = &lt;span class
="hljs-string">"eu-central-1"&lt;/span>
                &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class=
"hljs-keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.C
lient&lt;/span>).CreateSQSClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tClient.DeleteMessag
e(pQueueName, pReceiptHandle))

    }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

&lt;span class="hljs-keyword">ClassMethod&lt;/span> CreateQueue(pQueueName &lt;span c
lass="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>) &l
t;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.SQSSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
">as&lt;/span> &lt;span class="hljs-built_in">%External.Messaging.SQSClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSQSSettings &lt;span class="hljs-ke
yword">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.SQSQueueSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(&lt;span class="hljs-
keyword">##class&lt;/span>(AWS.Utils).GetCredentials(.tCredentials))
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.SQSSett
ings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.accessKey = tCredential
s(&lt;span class="hljs-string">"aws_access_key_id"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.secretKey = tCredential
s(&lt;span class="hljs-string">"aws_secret_access_key"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.sessionToken = tCredent
ials(&lt;span class="hljs-string">"aws_session_token"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.region = &lt;span class
="hljs-string">"eu-central-1"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateSQSClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)

        &lt;span class="hljs-keyword">set&lt;/span> tSQSSettings = &lt;span class="hl
js-keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.SQSQ
ueueSettings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()

        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.CreateQueue(pQueueName,tSQSSettings))
```

```
    }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

&lt;span class="hljs-keyword">ClassMethod&lt;/span> DeleteQueue(pQueueName &lt;span c
lass="hljs-keyword">As&lt;/span> &lt;span class="hljs-built_in">%String&lt;/span>) &l
t;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSettings &lt;span class="hljs-keywo
rd">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.SQSSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tClient &lt;span class="hljs-keyword
">as&lt;/span> &lt;span class="hljs-built_in">%External.Messaging.SQSClient&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSQSSettings &lt;span class="hljs-ke
yword">as&lt;/span> &lt;span class="hljs-
built_in">%External.Messaging.SQSQueueSettings&lt;/span>
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(&lt;span class="hljs-
keyword">##class&lt;/span>(AWS.Utils).GetCredentials(.tCredentials))
        &lt;span class="hljs-keyword">set&lt;/span> tSettings = &lt;span class="hljs-
keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.SQSSett
ings&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.accessKey = tCredential
s(&lt;span class="hljs-string">"aws_access_key_id"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.secretKey = tCredential
s(&lt;span class="hljs-string">"aws_secret_access_key"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.sessionToken = tCredent
ials(&lt;span class="hljs-string">"aws_session_token"&lt;/span>)
        &lt;span class="hljs-keyword">set&lt;/span> tSettings.region = &lt;span class
="hljs-string">"eu-central-1"&lt;/span>
        &lt;span class="hljs-keyword">set&lt;/span> tClient = &lt;span class="hljs-ke
yword">##class&lt;/span>(&lt;span class="hljs-built_in">%External.Messaging.Client&lt
;/span>).CreateSQSClient(tSettings.ToJSON(),.tSc)
        &lt;span class="hljs-built_in">$$$ThrowOnError&lt;/span>(tSc)

        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tClient.DeleteQueue(pQueueName))


    }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}

}
```

Ele contém métodos para criar e excluir filas, além de enviar e receber mensagens de uma fila.

Um dos principais pontos aqui é como autenticar. Não queria colocar minhas credenciais no meu código. Por isso, criei um método de ajuda para recuperar minhas credenciais no meu arquivo local e retornar como uma array subscrita para usar nos meus métodos da API:

```
&lt;span class="hljs-keyword">ClassMethod&lt;/span> GetCredentials(Output pCredential
s) &lt;span class="hljs-keyword">As&lt;/span> &lt;span class="hljs-
built_in">%Status&lt;/span>
{
    &lt;span class="hljs-keyword">#dim&lt;/span> tSc &lt;span class="hljs-keyword">as
&lt;/span> &lt;span class="hljs-built_in">%Status&lt;/span> = &lt;span class="hljs-
built_in">$$$OK&lt;/span>
    &lt;span class="hljs-keyword">set&lt;/span> tFilename = &lt;span class="hljs-
string">"/dur/.aws/credentials"&lt;/span>
    &lt;span class="hljs-keyword">try&lt;/span> {
        &lt;span class="hljs-keyword">set&lt;/span> tCredentialsFile = &lt;span class
="hljs-keyword">##class&lt;/span>(&lt;span class="hljs-built_in">%Stream.FileCharacte
r&lt;/span>).&lt;span class="hljs-built_in">%New&lt;/span>()
        &lt;span class="hljs-
built_in">$$$ThrowOnError&lt;/span>(tCredentialsFile.LinkToFile(tFilename))
        &lt;span class="hljs-comment">// first read the header&lt;/span>
        &lt;span class="hljs-
keyword">set&lt;/span> tBuffer = tCredentialsFile.ReadLine()
        &lt;span class="hljs-keyword">for&lt;/span> i=&lt;span class="hljs-number">1&
lt;/span>:&lt;span class="hljs-number">1&lt;/span>:&lt;span class="hljs-
number">3&lt;/span> {
            &lt;span class="hljs-
keyword">set&lt;/span> tBuffer = tCredentialsFile.ReadLine()
            &lt;span class="hljs-keyword">set&lt;/span> pCredentials(&lt;span class="
hljs-built_in">$piece&lt;/span>(tBuffer,&lt;span class="hljs-string">" ="&lt;/span>,&
lt;span class="hljs-number">1&lt;/span>)) = &lt;span class="hljs-built_in">$tr&lt;/sp
an>(&lt;span class="hljs-built_in">$piece&lt;/span>(tBuffer,&lt;span class="hljs-stri
ng">"= "&lt;/span>,&lt;span class="hljs-number">2&lt;/span>),&lt;span class="hljs-
built_in">$c&lt;/span>(&lt;span class="hljs-number">13&lt;/span>))
        }
    }
    &lt;span class="hljs-keyword">catch&lt;/span> tEx {
        &lt;span class="hljs-keyword">set&lt;/span> tSc = tEx.AsStatus()
    }

    &lt;span class="hljs-keyword">return&lt;/span> tSc
}
```

Para concluir este artigo, vamos criar uma fila ***community*** na região do AWS "eu-central-1" (Frankfurt, Alemanha).

A fila foi criada com sucesso e está visível no console do AWS para minha conta:

Em seguida, vamos enviar uma mensagem para essa fila:

O método call retorna 1. Portanto, a mensagem foi enviada com êxito.

Por fim, vamos usar o poll na fila do console do AWS:

A mensagem foi enviada para a fila com êxito.

Conclusão

A API de mensagens externas no InterSystems IRIS 2022.1 simplifica bastante a comunicação com as plataformas de streaming de eventos.
Espero que isso tenha sido útil.

#API #AWS #Implantação #InterSystems IRIS #InterSystems IRIS for Health

---

URL de origem:https://pt.community.intersystems.com/post/aproveitando-api-de-mensagens-externas-no-intersystems-iris%C2%A0