

# kafka with akka



kafka

kafka client

kafka Actor Akka Stream

.

kafka

[blocked URL](#)

AkkaStream , Graph-DSL

.

.

(throttle)

- 
- [DSL](#)

1.

:



kafka Java OS

kafka

## download

Link : <https://kafka.apache.org/downloads>

# Download

1.0.0 is the latest release. The current stable version is 1.0.0.

You can verify your download by following these [procedures](#) and using these [KEYS](#).

### 1.0.0

- Released November 1, 2017
- Source download: [kafka-1.0.0-src.tgz](#) (asc, sha512)
- Binary downloads:

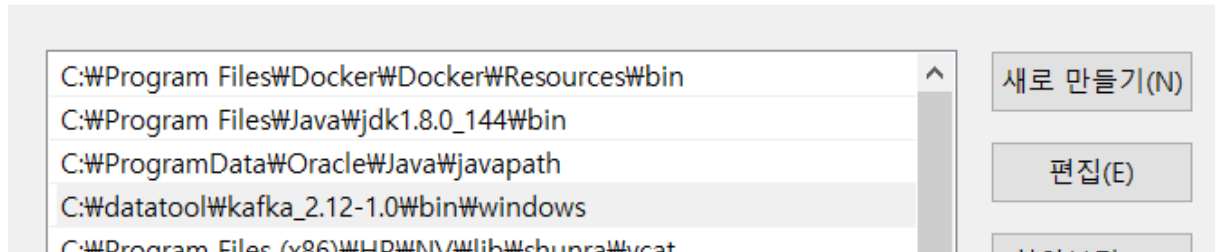
- Scala 2.11 - [kafka\\_2.11-1.0.0.tgz](#) (asc, sha512)
- Scala 2.12 - [kafka\\_2.12-1.0.0.tgz](#) (asc, sha512)

We build for multiple versions of Scala. This only matters if you are using Scala and you want a version built for the same Scala version you use. Otherwise any version should work (2.11 is recommended).

ex> C:\datatool\kafka\_2.12-1.0>

## path

환경 변수 편집



, path .

C:\datatool\kafka\_2.12-1.0\bin\windows

C:\datatool\kafka\_2.12-1.0> .

```
zookeeper-server-start config/zookeeper.properties
```

```
kafka-server-start config/server.properties
```

## Topic

kafka Topic .

```
kafka-topics --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test
```

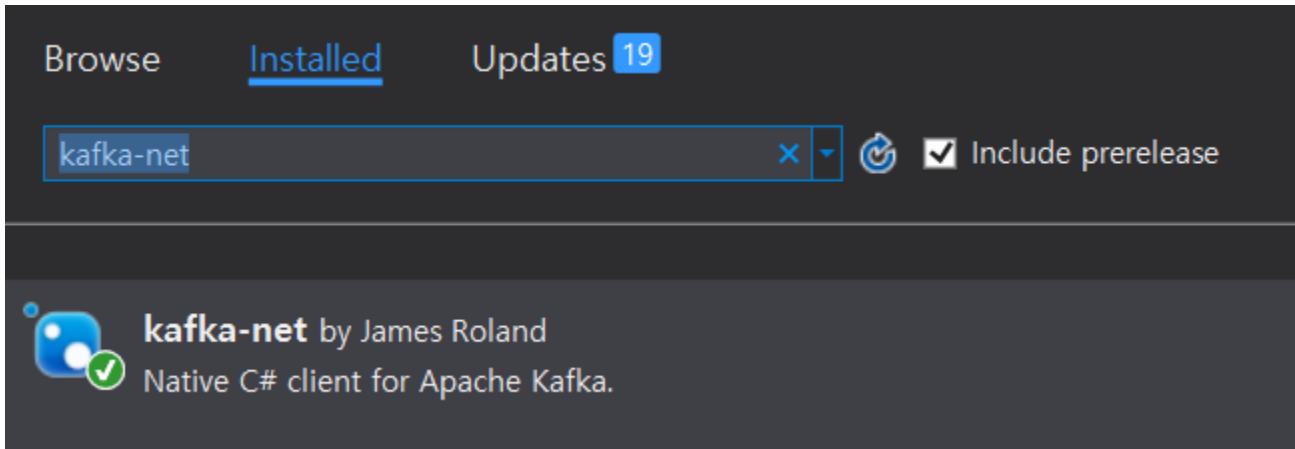
## Topic Console Listen

```
kafka-console-consumer --bootstrap-server localhost:9092 --topic test --from-beginning
```

KafkaTool UI .

<http://www.kafkatool.com/download.html>

## 2. (.net)



## KAFKA


### Producer

```
using KafkaNet;
var options = new KafkaOptions(new Uri("http://localhost:9092"), new Uri("http://localhost:9092"));
var router = new BrokerRouter(options);
var client = new Producer(router);
client.SendMessageAsync("test", new[] { new Message("hello world") }).Wait();
```

### Consumer

```
var options = new KafkaOptions(new Uri("http://localhost:9092"), new Uri("http://localhost:9092"));
var router = new BrokerRouter(options);
var consumer = new Consumer(new ConsumerOptions("TestHarness", router));

//Consume returns a blocking IEnumerable (ie: never ending stream)
foreach (var message in consumer.Consume())
{
    Console.WriteLine("Response: P{0},O{1} : {2}",
        message.Meta.PartitionId, message.Meta.Offset, message.Value);
}
```

 KAFKA (Producer)  
(Consumer)

## KAFKA AKKA.net

KAFKA

AKKA .

•

- (,)
- \



Apache Kafka( ) LinkedIn 2011

TPS .

[akka.net](#) kafka

kafka .

src : <http://git.webnori.com/projects/AKKA/repos/akkastudy/browse/Solution/ServiceB/STUDY/KafkaTest.cs>

- [At-Least-Once Delivery](#)
  - 
  -
- 
- 

## At-Least-Once Delivery

[At-Least-Once Delivery](#) Kafka

kafka, .

- at-most-once : .
- at-least-once : , , .
- exactly-once : , .

exactly-once , 100% .

TestCase :

- Kafka 20 .
- , Kafka .
- 10 , . .
- 20 .( )

```
actorSystem = ActorSystem.Create("ServiceB");
var kafkaActor = actorSystem.ActorOf<KafkaAtLeastOnceDeliveryReceiveActor>("kafka");
for(int i=0; i<20;i++ )
{
    kafkaActor.Tell("can you speak english? :" + i);
}
```

*Fire and forgot : ,*

... , .

```
. Tell
, Ask
,
, . - Future and Promise#futureactor
```

, At-Least-Once Delivery .

```
public class KafkaSendActor : ReceiveActor
{
    private ILoggingAdapter log = Context.GetLogger();
    protected int messageCnt = 0;
    private KafkaOptions kafka_options;
    private BrokerRouter kafka_router;
    private Producer client;
    private string topic = "test";

    public KafkaSendActor()
    {
        kafka_options = new KafkaOptions(new Uri("http://localhost:9092"), new Uri("http://localhost:9092"));
        kafka_router = new BrokerRouter(kafka_options);
        client = new Producer(kafka_router);
        IActorRef parentActor = null;

        Receive<Msg>(msg =>
        {
            parentActor = Sender;
            messageCnt++;
            //Test ...
            if ( messageCnt % 2 == 0 )
            {
                log.Warning("Test for Drop Message");
                return;
            }
            log.Debug("Yes I can");
            HandleProducer(msg);
        });

        Receive<Confirm>(msg =>
        {
            if( msg!=null)
                parentActor.Tell(msg, Self);
        });
    }

    public void HandleProducer( Msg msg)
    {
        Task.Run(async () =>
        {
            List<ProduceResponse> reponse = await client.SendMessageAsync( topic , new[] { new Message(msg.
Message) }));
            ProduceResponse resOne = reponse[0];
            Confirm confirm = new Confirm(msg.DeliveryId);
            if ( resOne.Error == 0 )
                return confirm;
            else
                return null;
        }).PipeTo(Self);
    }
}
```

Pipe

,  
.  
.

```
public class KafkaAtLeastOnceDeliveryReceiveActor : AtLeastOnceDeliveryReceiveActor
{
    private readonly IActorRef _destinationActor = Context.ActorOf<KafkaSendActor>();
    private ILoggingAdapter log = Context.GetLogger();

    public KafkaAtLeastOnceDeliveryReceiveActor()
    {
        Recover<MsgSent>(msgSent => Handler(msgSent));
        Recover<MsgConfirmed>(msgConfirmed => Handler(msgConfirmed));

        Command<string>(str =>
        {
            log.Debug("received:" + str);
            Persist(new MsgSent(str), Handler);
        });

        Command<Confirm>(confirm =>
        {
            // , .
            log.Debug("received confirm:" + confirm.DeliveryId);
            Persist(new MsgConfirmed(confirm.DeliveryId), Handler);
        });
    }

    private void Handler( MsgSent msgSent )
    {
        Deliver(_destinationActor.Path, l => new Msg(l, msgSent.Message));
    }

    private void Handler( MsgConfirmed msgConfirmed )
    {
        ConfirmDelivery(msgConfirmed.DeliveryId);
    }

    public override string PersistenceId { get; } = "persistence-id";
}
```

. AtLeastOnceDeliveryReceiveActor

.  
.

AtLeastOnceDelivery () .

.

- ?
- 
- (resOne.Error)
- ( , )

: <https://doc.akka.io/docs/akka/current/persistence.html#at-least-once-delivery> (AKKA SCALA )

```
file:///D:/work/hobby/git.webnori.com/akkastudy/Solution/ServiceB/bin/Debug/ServiceB.EXE
[DEBUG][2017-12-04 오전 7:32:08][Thread 0044][akka://ServiceB/user/kafka/$a#1140367166]] Yes I can
[WARNING][2017-12-04 오전 7:32:08][Thread 0044][akka://ServiceB/user/kafka/$a#1140367166]] Test for Drop Message
[DEBUG][2017-12-04 오전 7:32:08][Thread 0042][akka://ServiceB/user/kafka/$a#1140367166]] Yes I can
[DEBUG][2017-12-04 오전 7:32:08][Thread 0012][akka://ServiceB/user/kafka#1843957568]] received confirm:4
[DEBUG][2017-12-04 오전 7:32:08][Thread 0042][akka://ServiceB/user/kafka#1843957568]] received confirm:12
[DEBUG][2017-12-04 오전 7:32:08][Thread 0042][akka://ServiceB/user/kafka#1843957568]] received confirm:20
[WARNING][2017-12-04 오전 7:32:13][Thread 0044][akka://ServiceB/user/kafka/$a#1140367166]] Test for Drop Message
[DEBUG][2017-12-04 오전 7:32:13][Thread 0042][akka://ServiceB/user/kafka/$a#1140367166]] Yes I can
[DEBUG][2017-12-04 오전 7:32:13][Thread 0042][akka://ServiceB/user/kafka#1843957568]] received confirm:16
[WARNING][2017-12-04 오전 7:32:18][Thread 0042][akka://ServiceB/user/kafka/$a#1140367166]] Test for Drop Message
[DEBUG][2017-12-04 오전 7:32:23][Thread 0032][akka://ServiceB/user/kafka/$a#1140367166]] Yes I can
[DEBUG][2017-12-04 오전 7:32:23][Thread 0044][akka://ServiceB/user/kafka#1843957568]] received confirm:8

can you speak english? :0
can you speak english? :2
can you speak english? :4
can you speak english? :6
can you speak english? :8
can you speak english? :10
can you speak english? :12
can you speak english? :14
can you speak english? :16
can you speak english? :18
can you speak english? :1
can you speak english? :5
can you speak english? :13
can you speak english? :17
can you speak english? :9
can you speak english? :3
can you speak english? :11
can you speak english? :19
can you speak english? :15
can you speak english? :7
```

... ()

Akka Stream Throttle , ()

(sink,kafka) .

? ( )

Sleep .( ,)

AkkaStream .

1

, Throttle .

```
ActorSystem actorSystem;
ActorMaterializer materializer;
KafkaOptions kafka_options;
BrokerRouter kafka_router;

kafka_options = new KafkaOptions(new Uri("http://localhost:9092"), new Uri("http://localhost:9092"));
kafka_router = new BrokerRouter(kafka_options);
var client = new Producer(kafka_router);

actorSystem = ActorSystem.Create("ServiceB");
var materializer = actorSystem.Materializer();

Source<int, NotUsed> source = Source.From(Enumerable.Range(1, 10));
var factorials = source.Scan(1, ( acc, next ) => acc + next);

factorials
    .Throttle(1, TimeSpan.FromSeconds(1), 1, ThrottleMode.Shaping)
    .RunForeach(i => {
        Console.WriteLine("SendOne");
        client.SendMessageAsync("test", new[] { new Message(i.ToString()) });
    }, materializer);
```

- ( + )
- : ( )

## KAKFA AKKA(JAVA/SCALA)

akka-stream-kafka akka stream kafka client

. stream

.

: <https://doc.akka.io/docs/akka-stream-kafka/current/producer.html>



Akka Streams Reactive Streams , AKKA

Reactive Streams .

<http://www.reactive-streams.org/>