

# Streams Quickstart Guide

Stream

```
using Akka;
using Akka.Streams;
using Akka.Streams.Dsl;

actorSystem = ActorSystem.Create("ServiceB");
Source<int, NotUsed> source = Source.From(Enumerable.Range(1, 100));
using (var materializer = actorSystem.Materializer())
{
    source.RunForeach(i => Console.WriteLine(i.ToString()), materializer);
    Console.WriteLine("Loop    ");
}
```

1 100 .

Loop

```
for(int srcNum = 0; srcNum < 100; srcNum++)
    Console.WriteLine(srcNum.ToString());

Console.WriteLine("Loop    ");
```

.
AkkaStream .

?
AKKA Stream .
AkkaStream .

.
Loop int.Max . Loop ,
.

AKKA
• Source : , ( )
• Materializer : ( )

```

using System.IO;
using Akka.IO;

actorSystem = ActorSystem.Create("ServiceB");
Source<int, NotUsed> source = Source.From(Enumerable.Range(1, 100));
using (var materializer = actorSystem.Materializer())
{
    source.RunForeach(i => Console.WriteLine(i.ToString()), materializer);

    var factorials = source.Scan( 1 , (acc, next) => acc + next);
    var result =
        factorials
            .Select(num => ByteString.FromString($"{num}\n"))
            .RunWith(FileIO.ToFile(new FileInfo("factorials.txt")), materializer);
}

```

1-100 .

- Scan . ( 1~100 )
- Select .
- RunWith (Sink .) , .
- RunWith IOResult IO/ .

Akka

- Sink: ()
- : Sink

## 2-Reusable Pices

```

using Akka.Streams.IO;

public static Sink<string, Task<IOResult>> LineSink(string filename)
{
    return Flow.Create<string>()
        .Select(s => ByteString.FromString($"{s}\n"))
        .ToMaterialized(FileIO.ToFile(new FileInfo(filename)), Keep.Right);
}

public void StreamTest2()
{
    actorSystem = ActorSystem.Create("ServiceB");
    Source<int, NotUsed> source = Source.From(Enumerable.Range(1, 100));
    using (var materializer = actorSystem.Materializer())
    {
        source.RunForeach(i => Console.WriteLine(i.ToString()), materializer);

        var factorials = source.Scan(1, (acc, next) => acc + next);
        factorials.Select(_ => _.ToString()).RunWith(LineSink("factorials.txt"), materializer);
    }
}

```

Akka Stream

- Sink , Sink .
- , ( Keep.Right) Flow .

Akka

- Flow :

### 3-TimeBased Processing

```
actorSystem = ActorSystem.Create("ServiceB");
Source<int, NotUsed> source = Source.From(Enumerable.Range(1, 100));
using (var materializer = actorSystem.Materializer())
{
    var factorials = source.Scan(1, (acc, next) => acc + next);
    factorials
        .ZipWith(Source.From(Enumerable.Range(0, 100)), (num, idx) => $"{idx}! = {num}")
        .Throttle(1, TimeSpan.FromSeconds(1), 1, ThrottleMode.Shaping)
        .RunForeach(i => Console.WriteLine(i.ToString()), materializer).Wait();
}
```

(throttle) 1

1 .

sleep()

AkkaStream ,

RunnableGraph .