

Öret

Kronan

## TPL Dataflow

Pär Nordström, Valtech

Frågor?

Tar vi på en gång...

På slutet...

Eller senare

## Valtech Tech Day 2010...

```
let isPrime (n:int) =
    let bound = int (System.Math.Sqrt(float n))
    seq {2 .. bound} |> Seq.exists (fun x -> n % x = 0) |> not


let primeAsync n =
    async { return (n, isPrime n) }

let primes m n =
    seq {m .. n}
    |> Seq.map primeAsync
    |> Async.Parallel
    |> Async.RunSynchronously
    |> Array.filter snd
    |> Array.map fst

primes 0 1000 |> Array.iter (printfn "%d")
```

”F# har stora fördelar  
vid asynkrona flöden”

# TPL Dataflow, DevLabs


DevLabs Search MSDN with Bing 

[Home](#) [About](#) [Projects](#) [Forums](#)

[Casablanca](#) [Debugger Canvas](#) [Code Contracts](#) [Solver Foundation](#) [Sho](#) [TPL Dataflow](#)

---

## TPL Dataflow



### TPL Dataflow

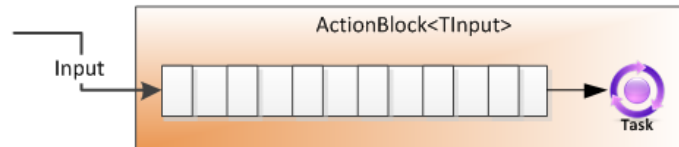
```
Distribute {  
    // Join th  
    IEnumerabl  
}
```

```
    public s  
    public s  
    public i
```

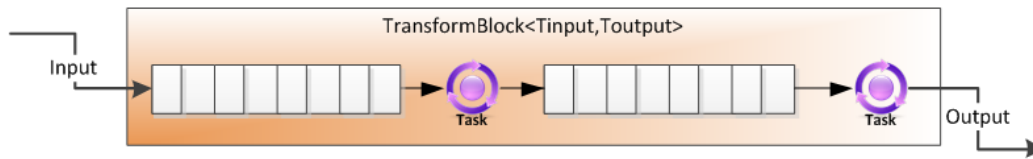
```
Console.Wr:  
EnumerateF:  
Console.Wr:  
EnumerateF:
```

Harnessing boundless capacity across the client, cluster and cloud

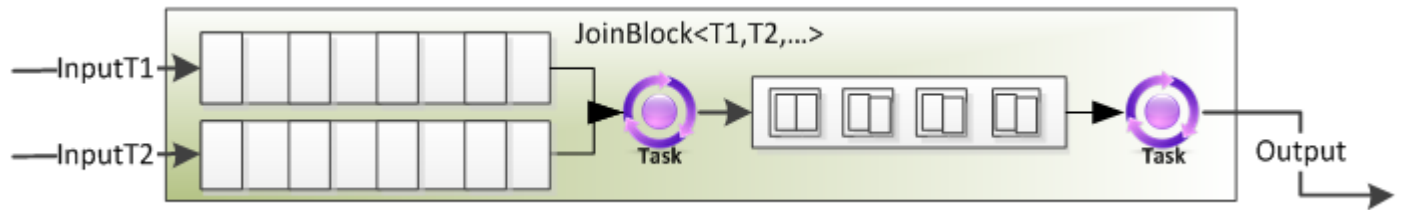
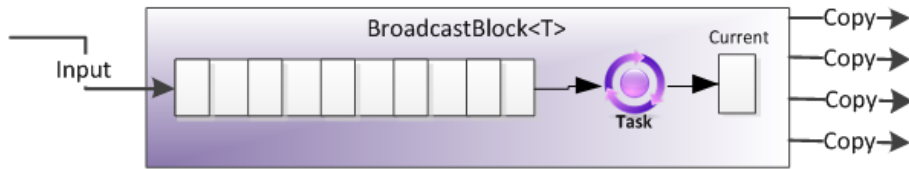
ActionBlock<T>



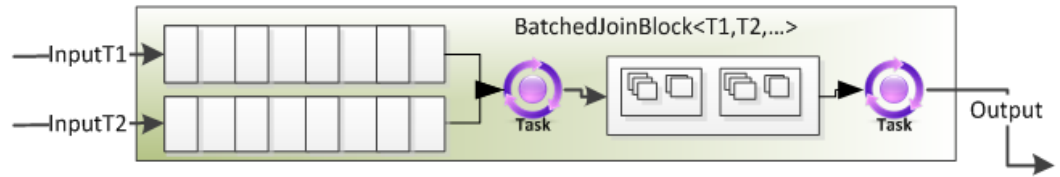
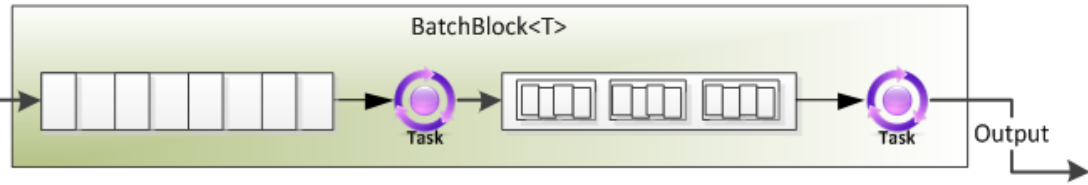
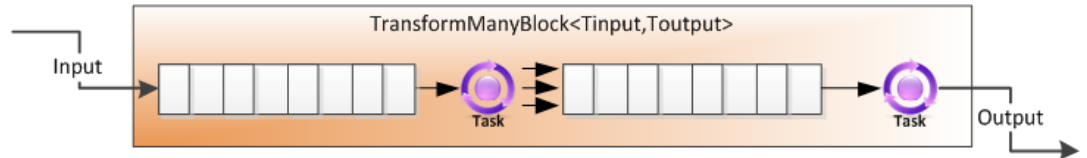
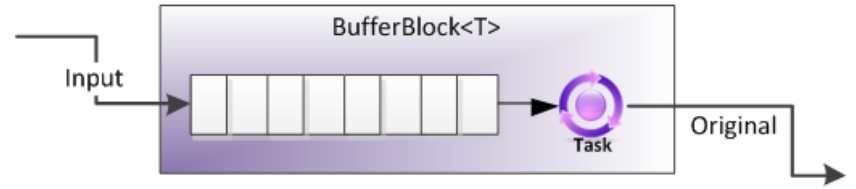
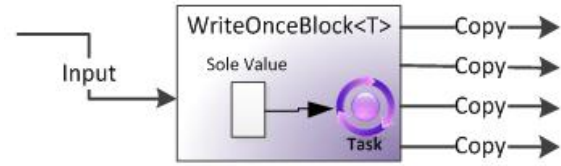
TransformBlock<Tin,TOut>



BroadcastBlock<T>  
JoinBlock<T1,T2,...>



Fler...





# Asynkrona flöden

```
let isPrime (n:int) =
    let bound = int (System.Math.Sqrt(float n))
    seq {2 .. bound} |> Seq.exists (fun x -> n % x = 0) |> not

let primeAsync n =
    async { return (n, isPrime n) }

let primes m n =
    seq {m .. n}
    |> Seq.map primeAsync
    |> Async.Parallel
    |> Async.RunSynchronously
    |> Array.filter snd
    |> Array.map fst

primes 0 1000 |> Array.iter (printfn "%d")
```

```
private static bool IsPrime(int n)
{
    var bound = Convert.ToInt32(Math.Floor(Math.Sqrt(n)));
    return bound >= 2 ? !Enumerable.Range(2, bound - 1).Any(x => ((n % x) == 0)) : true;
}

private static Tuple<int, bool> PrimeAsync(int n)
{
    return new Tuple<int, bool>(n, IsPrime(n));
}

private static ISourceBlock<int> Primes(int min, int max)
{
    return Enumerable.Range(min, max - min)
        .Map(PrimeAsync)
        .Filter(t => t.Item2)
        .Map(t => t.Item1);
}

static void Main(string[] args)
{
    Primes(0, 100000).Iter(n => Console.WriteLine(n));
}
```

---

# Frågor

---

- @parnordstrom
- parnordstrom.com