

1 System.Threading.Parallel.ParallelForEach<T 2 > Class

```
3 [ILAsm]  
4 .class public sealed serializable ParallelForEach<T> extends  
5 System.Threading.Parallel.ParallelLoop<!0>  
6 [C#]  
7 public sealed class ParallelForEach<T>: ParallelLoop<T>
```

8 Assembly Info:

- 9 • *Name:* System.Threading.Parallel
- 10 • *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- 11 • *Version:* 2.0.x.x
- 12 • *Attributes:*
 - 13 ○ CLSCompliantAttribute(true)

14 Summary

15 A parallel loop over a collection containing types of T.

16 **Inherits From:** System.Threading.Parallel.ParallelLoop<T>

17

18 **Library:** Parallel

19

20 **Thread Safety:** All public static members of this type are safe for multithreaded operations.
21 No instance members are guaranteed to be thread safe.

22

23 Description

24 A System.Threading.Parallel.ParallelForEach<T> iterates over an enumerable
25 collection. Method System.Threading.Parallel.ParallelForEach<T>.BeginRun
26 activates processing of the iterations, using a callback provided. The collection shall not
27 change while the System.Threading.Parallel.ParallelForEach<T> is active,
28 otherwise the behavior is undefined. Inherited method
29 System.Threading.Parallel.ParallelLoop<T>.EndRun blocks until all iterations are
30 finished. Inherited method System.Threading.Parallel.ParallelLoop<T>.Run is
31 shorthand for System.Threading.Parallel.ParallelForEach<T>.BeginRun and
32 System.Threading.Parallel.ParallelLoop<T>.EndRun.

33

34 [*Note:* System.Threading.Parallel.ParallelForEach<T> is generally none-scalable in
35 terms of parallelism, because the enumerator is inherently sequential. If the collection
36 allows random access, consider using class System.Threading.Parallel.ParalleFor
37 instead.]

38

39

40

1
2 **ParallelForEach<T> (System.IEnumerable<T>**
3 **, System.Int32) Constructor**

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(class  
6 System.Collections.Generic.IEnumerable<T> collection, int32 numThreads)  
  
7 [C#]  
8 public ParallelForEach(IEnumerable<T> collection, int numThreads)
```

9 **Summary**

10 Constructs a `System.Threading.Parallel.ParallelForEach<T>` for iterating over a
11 collection.

12 **Parameters**

Parameter	Description
<i>collection</i>	collection of values over which to iterate
<i>numThreads</i>	maximum number of threads to use

13
14 **Description**

15 The loop does not start executing until at least method
16 `System.Threading.Parallel.ParallelForEach<T>.BeginRun` is called and possibly not
17 until method `System.Threading.Parallel.ParallelLoop<T>.EndRun` is called.
18
19 If `numThreads` is 0, then up to
20 `System.Threading.Parallel.ParallelEnvironment.MaxThreads` threads are used
21 instead. The value includes the thread that created the
22 `System.Threading.Parallel.ParallelFor<T>`, hence using `numThreads=1` causes
23 sequential execution.

24 **Exceptions**

Exception	Condition
System.ArgumentException	The value for <code>numThreads</code> is negative

1

2 ParallelForEach<T>.BeginRun(System.Action 3 <T>) Method

```

4 [ILAsm]
5 .method public hidebysig override void BeginRun(class System.Action<!0>
6 action)
7
8 [C#]
9 public override void BeginRun(Action<T> action)

```

9 Summary

10 Begin executing iterations.

11 Parameters

Parameter	Description
<i>action</i>	The System.Delegate that processes each work item.

12

13 Description

14 This method is not thread safe. It should be called only once for a given instance of a
15 System.Threading.Parallel.ParallelWhile<T>.

16

17 [*Note:* Implementations, particularly on single-threaded hardware, are free to employ
18 the calling thread to execute all loop iterations.]

19

20

21 Exceptions

Exception	Condition
System.ArgumentNullException	<i>action</i> is null.

22

23

1 ParallelForEach<T>.Cancel() Method

```
2 [ILAsm]  
3 .method public hidebysig override void Cancel()  
4 [C#]  
5 public override void Cancel()
```

6 Summary

7 Cancel any iterations that have not yet started

8 Description

9 This method is safe to call concurrently on the same instance.

10

11 Does not cancel any future iterations that might be added.

12