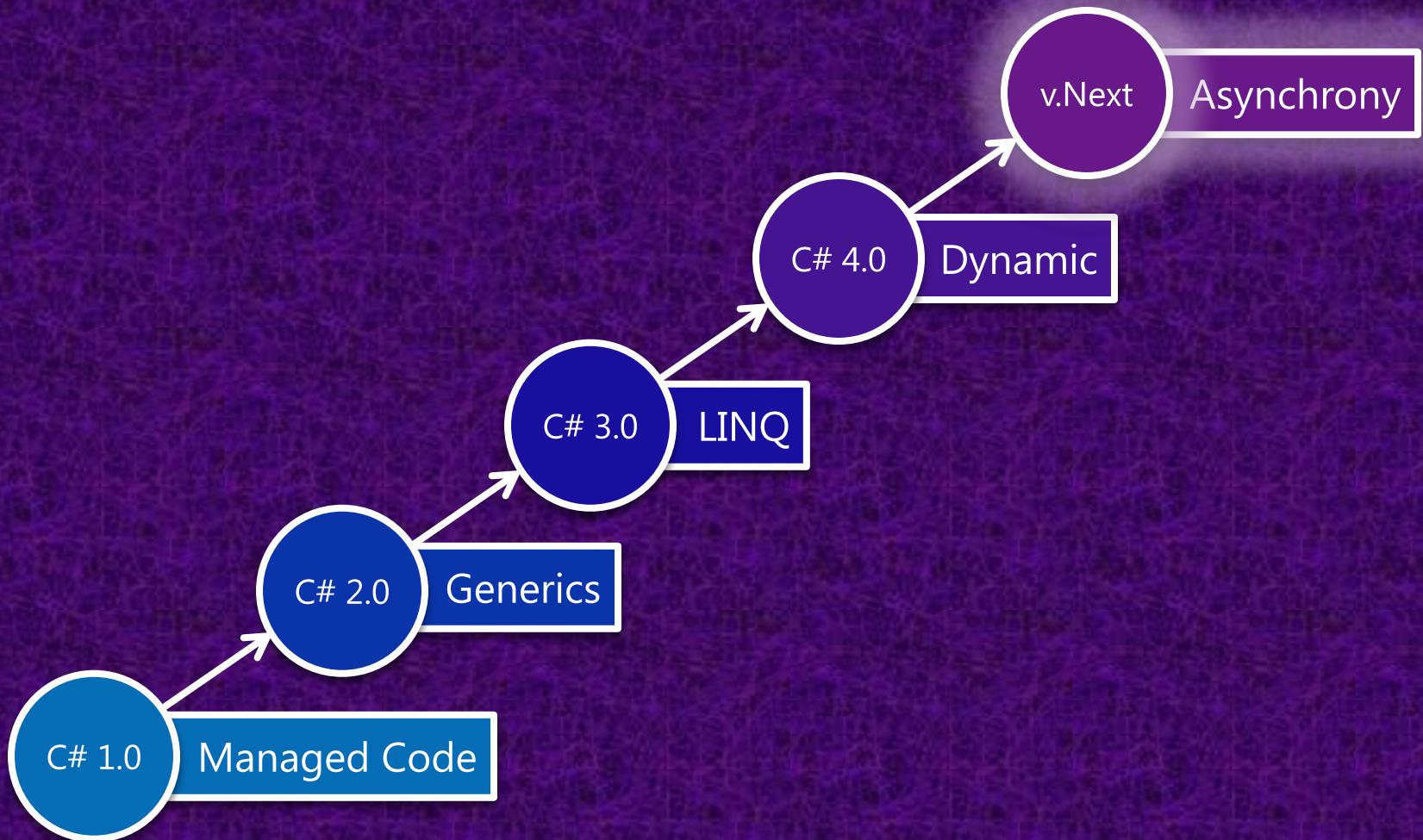


# Asynchronous Programming in .NET

Mads Torgersen  
C# Language PM  
Microsoft

# C# Evolution



# Synchronous vs. Asynchronous

```
var data = DownloadData(...);  
ProcessData(data);
```



```
DownloadDataAsync(..., data => {  
    ProcessData(data);  
});
```



# Synchronous vs. Asynchronous

```
var data = DownloadData(...);  
ProcessData(data);
```



```
DownloadDataAsync(..., data => {  
    ProcessData(data);  
});
```



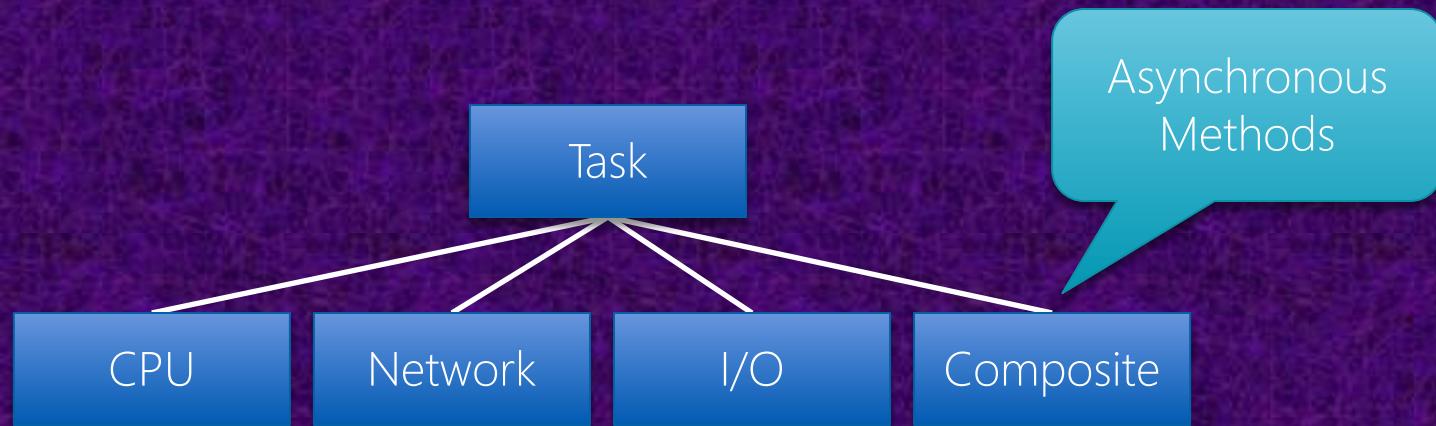
DownloadDataAsync

ProcessData

# Asynchronous Responsive UI

## demo

# Unifying Asynchrony



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



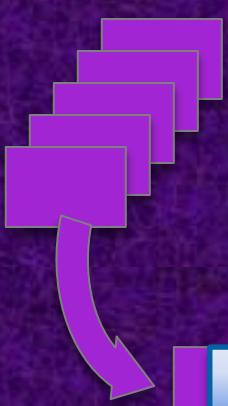
Message Pump

UI Thread

# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

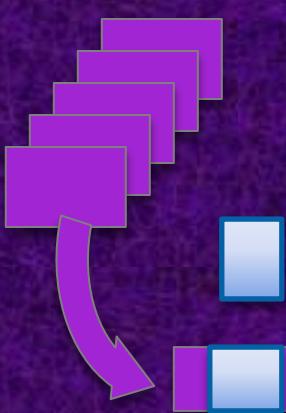
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

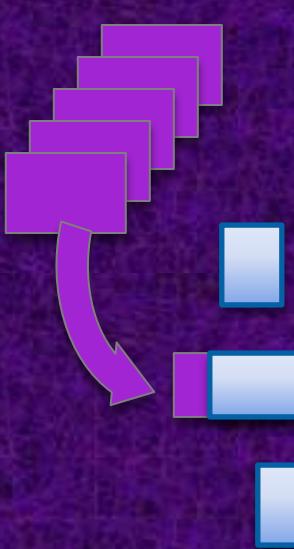
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

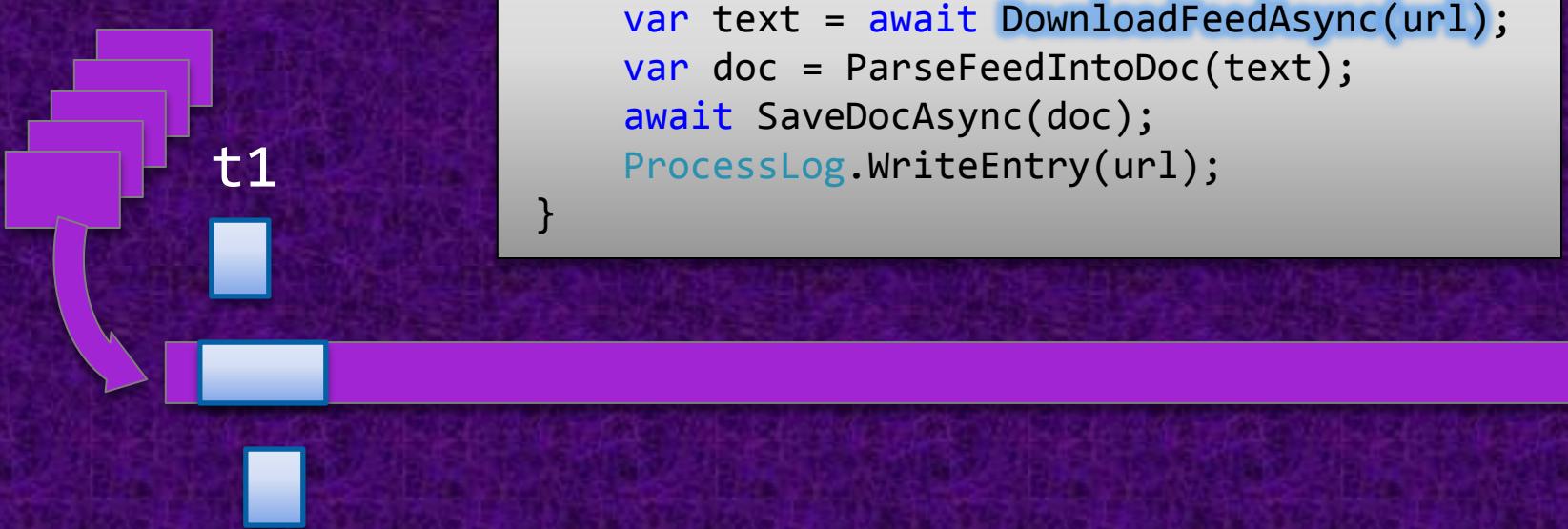
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

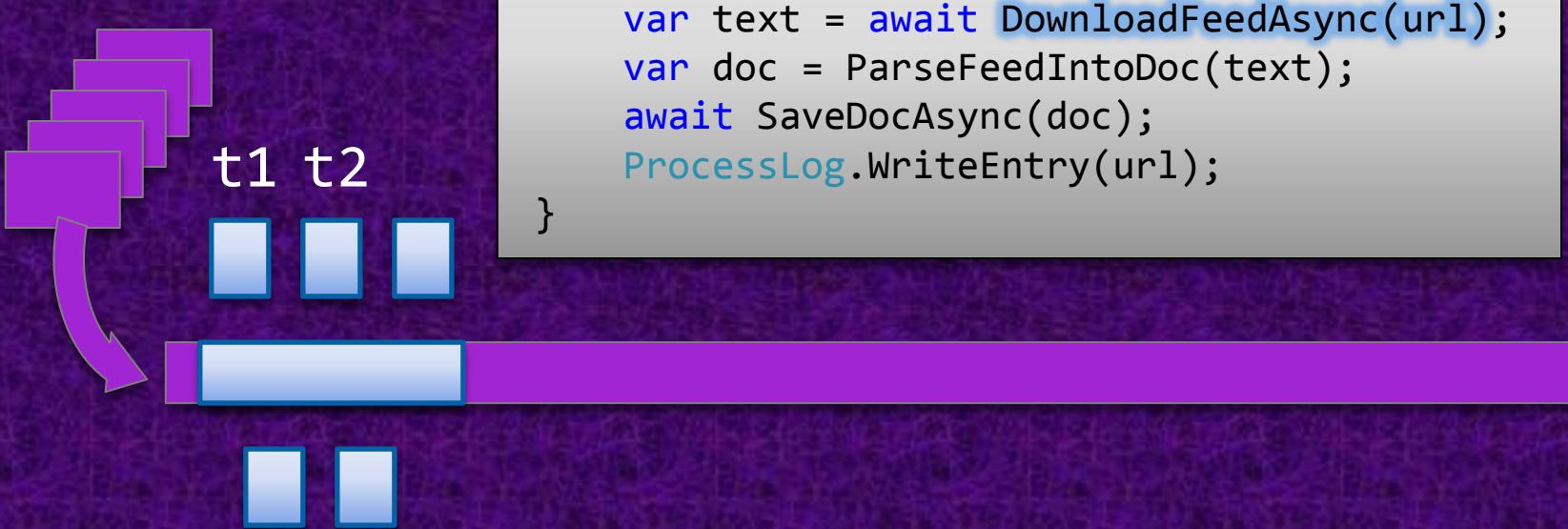
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

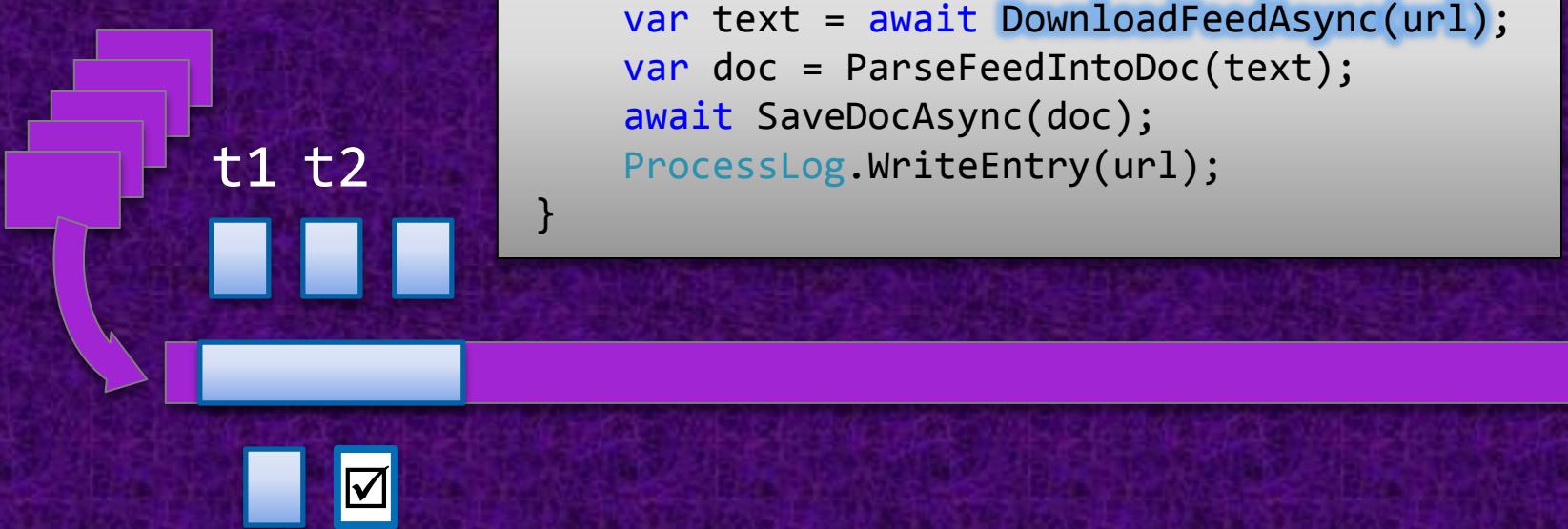
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

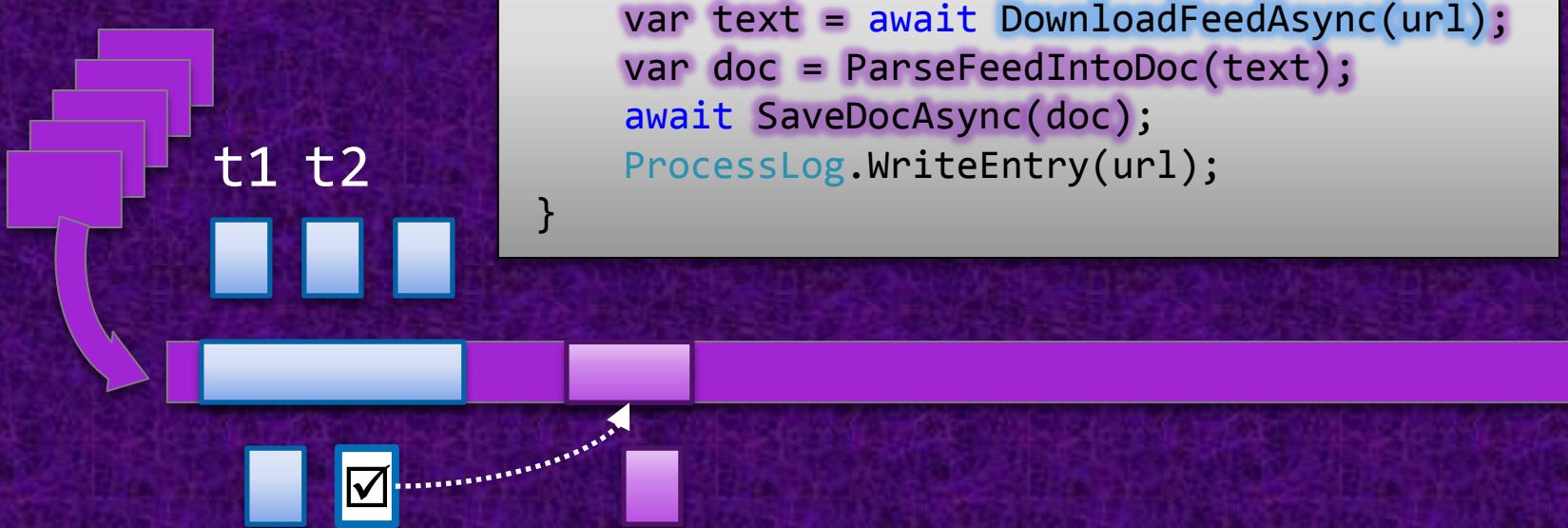
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

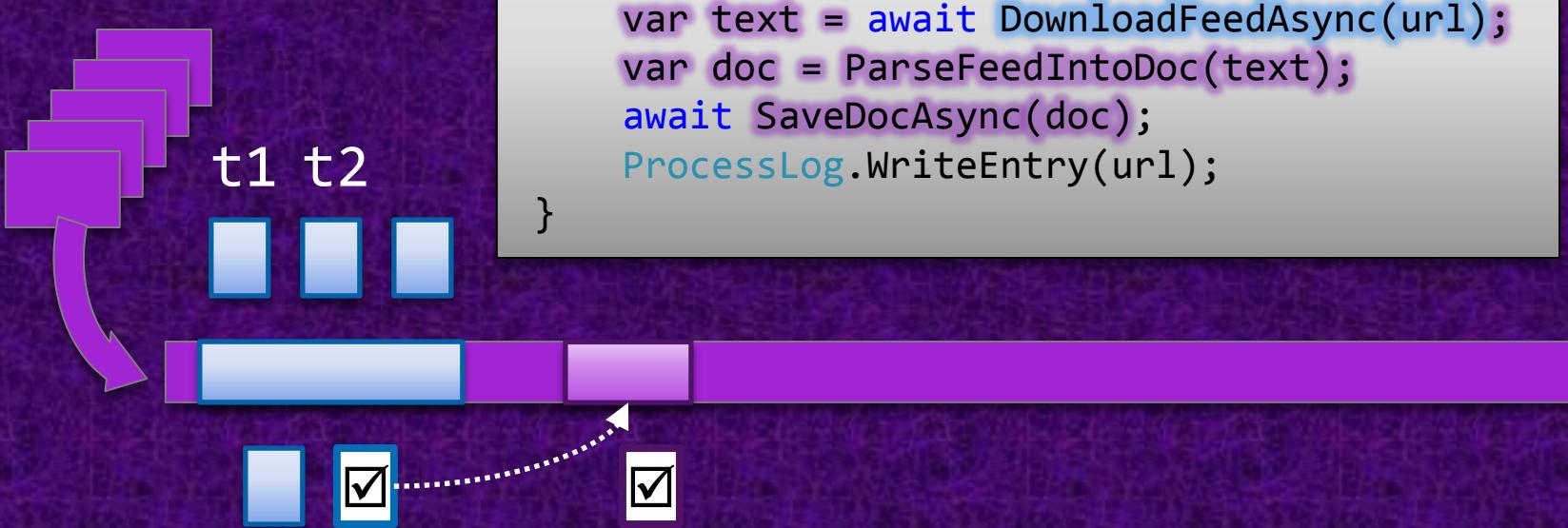
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

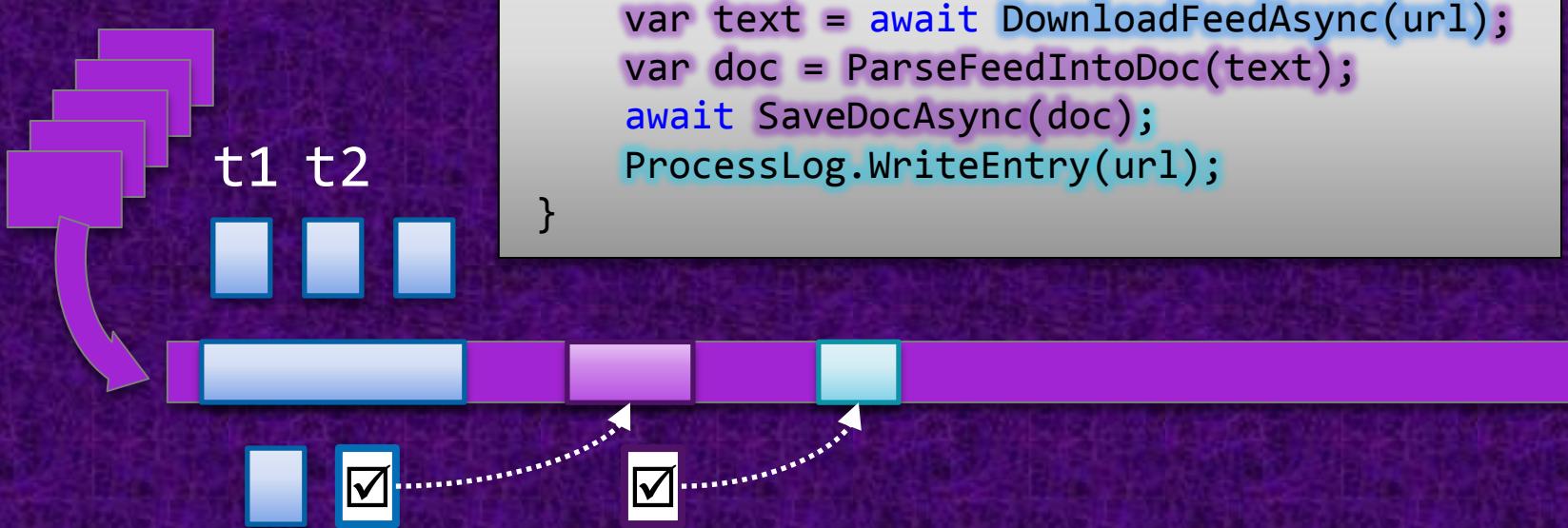
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

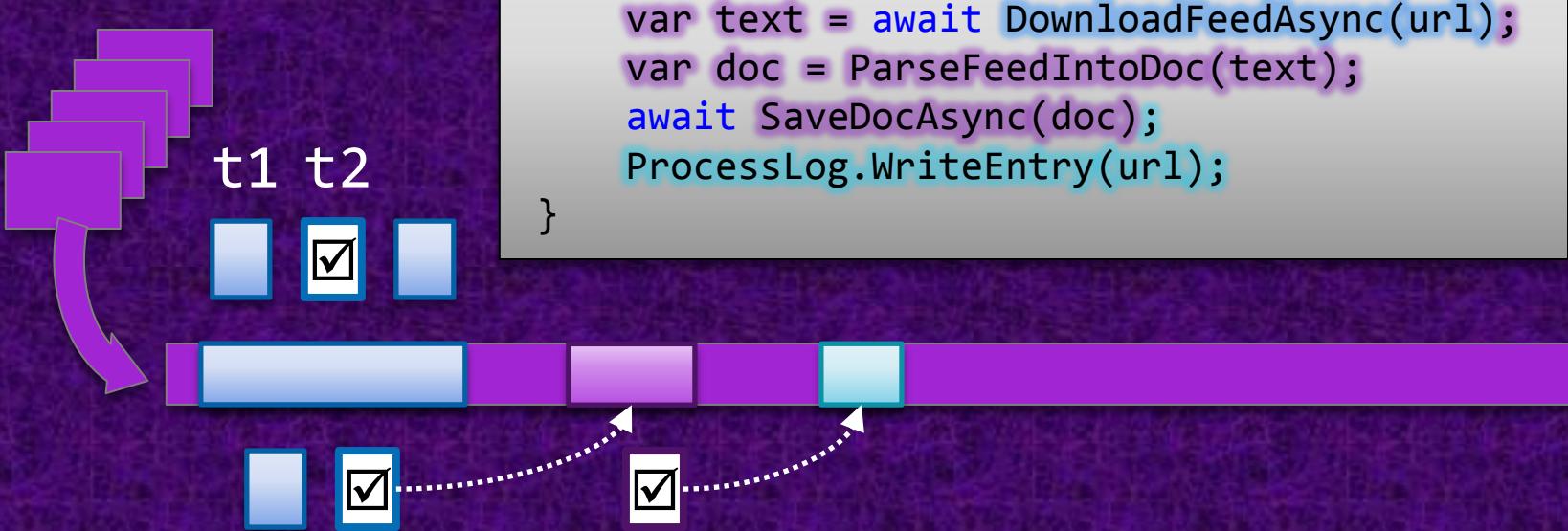
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

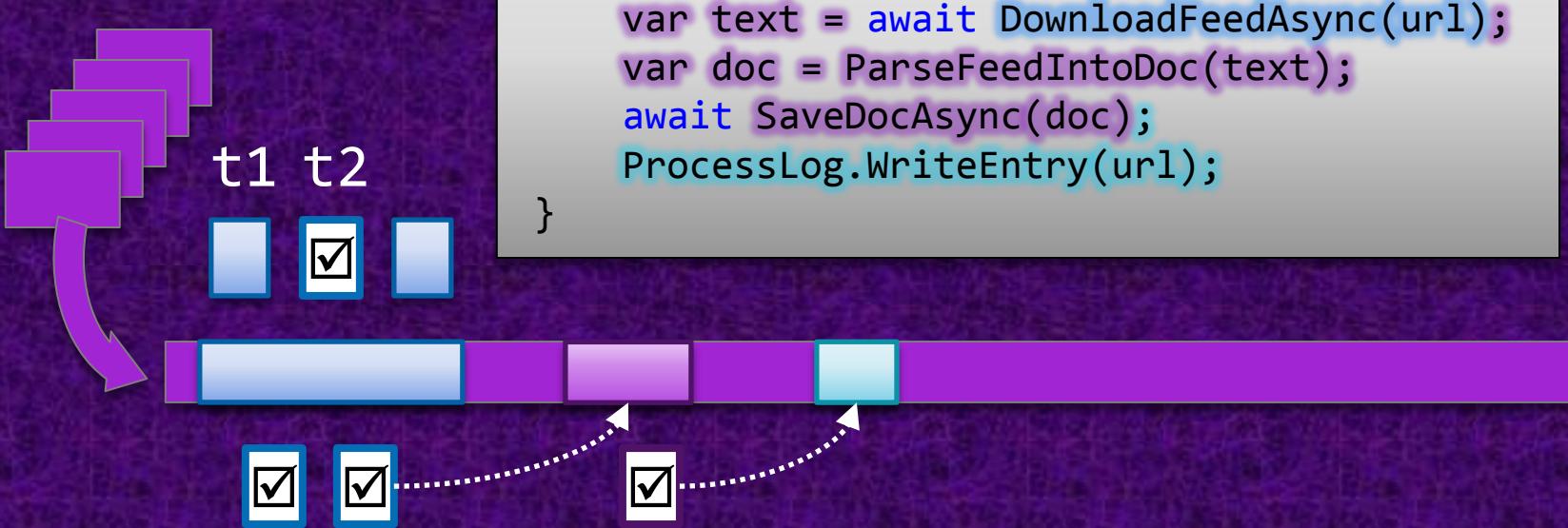
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

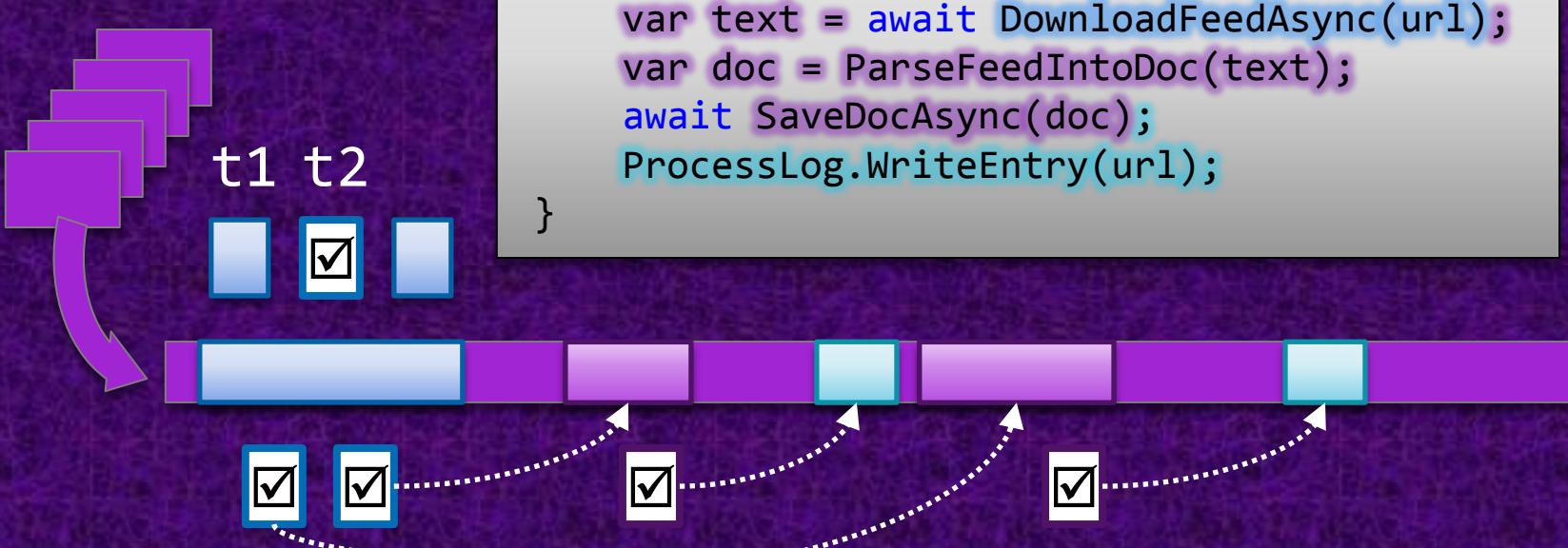
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

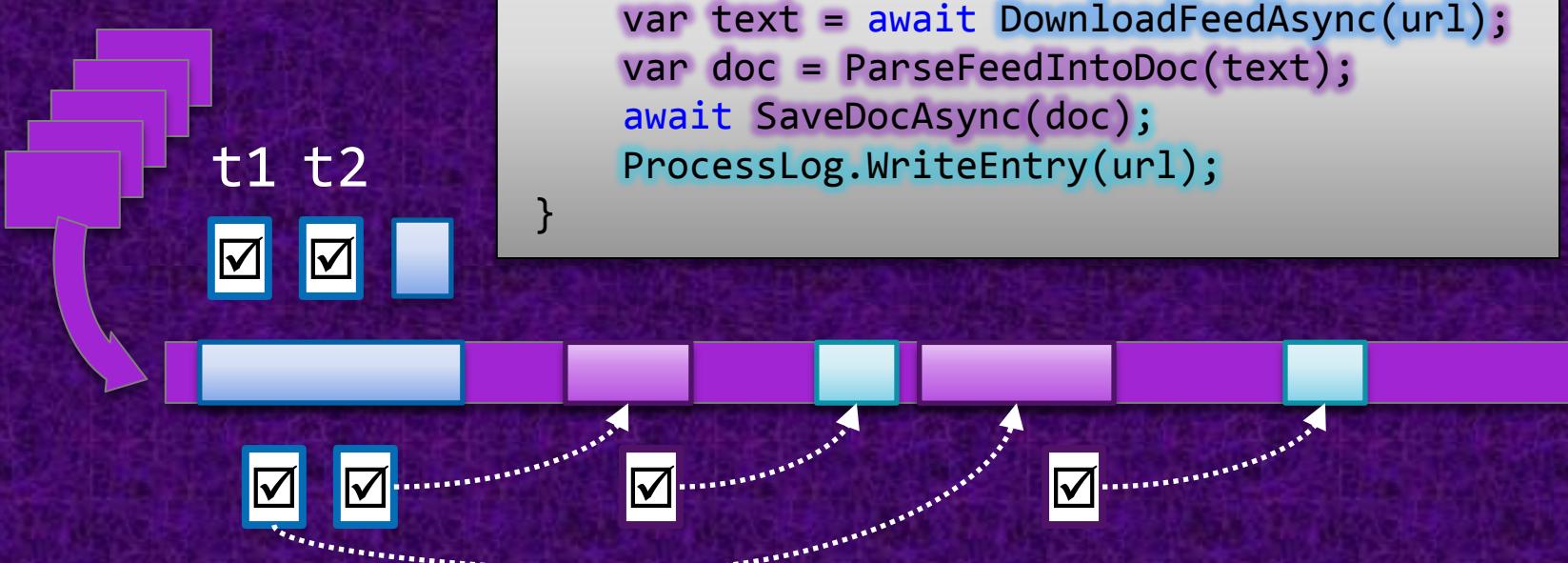
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

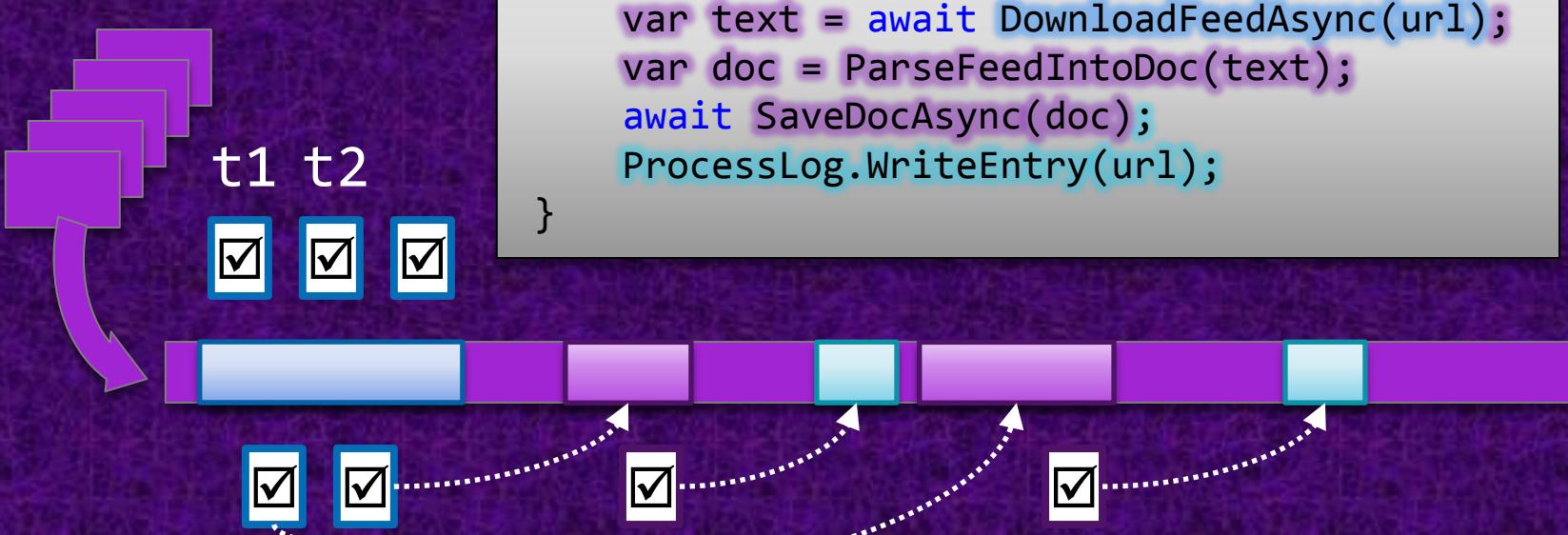
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {  
    var t1 = ProcessFeedAsync("www.acme.com/rss");  
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");  
    await Task.WhenAll(t1, t2);  
    DisplayMessage("Done");  
}
```

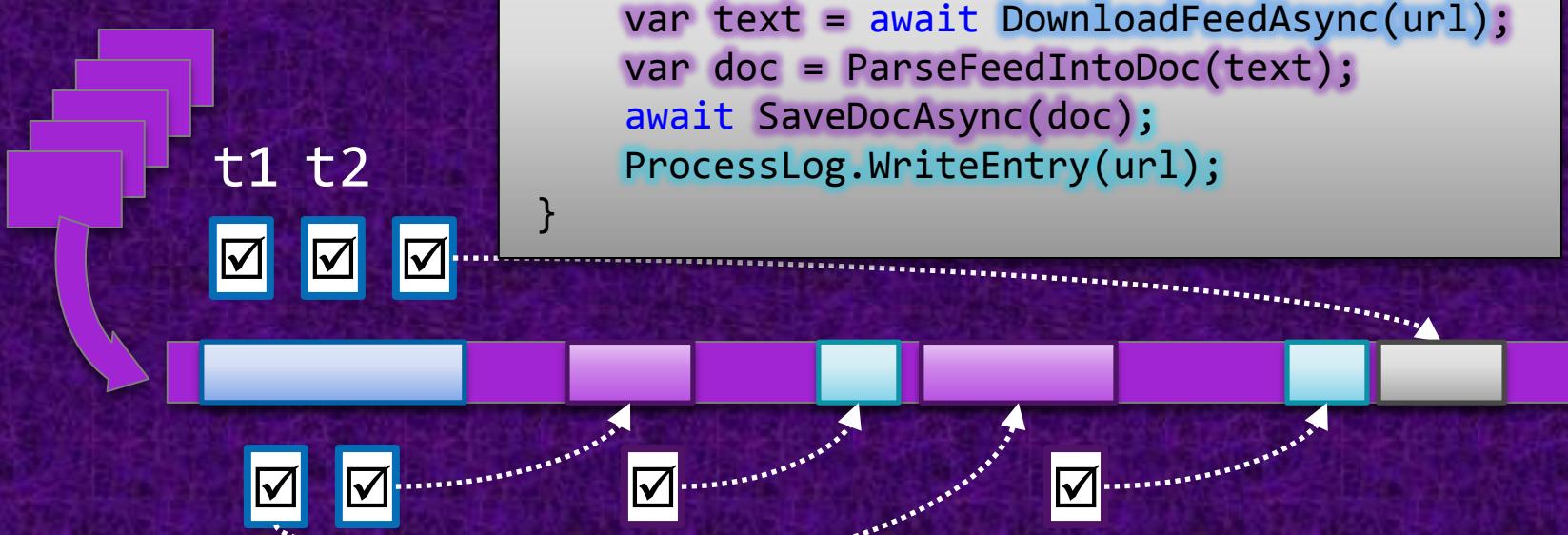
```
async Task ProcessFeedAsync(string url) {  
    var text = await DownloadFeedAsync(url);  
    var doc = ParseFeedIntoDoc(text);  
    await SaveDocAsync(doc);  
    ProcessLog.WriteEntry(url);  
}
```



# Asynchronous Control Flow

```
async void DoWorkAsync() {
    var t1 = ProcessFeedAsync("www.acme.com/rss");
    var t2 = ProcessFeedAsync("www.xyznews.com/rss");
    await Task.WhenAll(t1, t2);
    DisplayMessage("Done");
}
```

```
async Task ProcessFeedAsync(string url) {
    var text = await DownloadFeedAsync(url);
    var doc = ParseFeedIntoDoc(text);
    await SaveDocAsync(doc);
    ProcessLog.WriteEntry(url);
}
```



# How Does it Work?

```
async Task< XElement > GetRssAsync(string url) {  
    var client = new WebClient();  
    var task = client.DownloadStringTaskAsync(url);  
    var text = await task;  
    var xml = XElement.Parse(text);  
    return xml;  
}
```

# How Does it Work? (Sort of)

```
async Task< XElement > GetRssAsync(string url) {
    var client = new WebClient();
    var task = client.DownloadStringTaskAsync(url);
    return task.ContinueWith(t => {
        var text = t.Result;
        var xml = XElement.Parse(text);
        return xml;
    });
}
```

# How Does it Work? (Really)

```
async Task< XElement > GetRssAsync(string url) {  
    var client = new WebClient();  
    var task = client.DownloadStringTaskAsync(url);  
    var text = await task;  
    var xml = XElement.Parse(text);  
    return xml;  
}
```

# How Does it Work?

```
Task< XElement > GetRssAsync(string url) {
    var $builder = AsyncTaskMethodBuilder< XElement >.Create();
    var $state = 0;
    TaskAwaiter< string > $a1;
    Action $resume = delegate {
        try {
            if ($state == 1) goto L1;
            var client = new WebClient();
            var task = client.DownloadStringTaskAsync(url);
            $state = 1;
            $a1 = task.GetAwaiter();
            if (!$a1.IsCompleted) return;
            $a1.OnCompleted($resume);
        L1: var text = $a1.GetResult();
            var xml = XElement.Parse(text);
            $builder.SetResult(xml);
        }
        catch (Exception $ex) { $builder.SetException($ex); }
    };
    $resume();
    return $builder.Task;
}
```

# Awaiting a Task

```
public class Task<TResult> : Task
{
    ...
    public TaskAwaiter<TResult> GetAwaiter();
}

public struct TaskAwaiter<TResult>
{
    public bool IsCompleted { get; }
    public void OnCompleted(Action a);
    public TResult GetResult();
}
```

# Awaiting a Task

```
public class Task<TResult> : Task
{
    ...
    public TaskAwaiter<TResult> GetAwaiter();
}

public struct TaskAwaiter<TResult>
{
    public bool IsCompleted { get; }
    public void OnCompleted(Action a)
    {
        var sc = SynchronizationContext.Current;
        task.ContinueWith(_ => sc.Post(a));
    }
    public TResult GetResult();
}
```

# Asynchronous Methods

No more callbacks!

<http://msdn.com/vstudio/async>



© 2011 Microsoft Corporation. All rights reserved. Microsoft, Visual Studio, the Visual Studio logo, and [list other trademarks referenced] are trademarks of the Microsoft group of companies.

The information herein is for informational purposes only and represents the current view of Microsoft Corporation as of the date of this presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information provided after the date of this presentation.

MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS PRESENTATION.