

加速下载体验：C#多线程分块下载文件与实时进度展示

作者：微信公众号：【架构师老卢】

12-1 20:49

128



概述：该C#示例演示了如何使用多线程分块下载文件并显示下载进度。程序通过确定文件大小，创建多个线程，分配下载范围，同时下载文件块，最后合并文件。通过简单的控制台应用，用户可以清晰地看到下载进度。此方法提高了下载效率，更好地利用了网络带宽。

多线程分块下载文件的原理是将文件分成多个块，每个线程负责下载一个块的数据，最后将所有块合并成完整的文件。这样可以提高下载速度，并充分利用网络带宽。

方法与步骤

- 确定下载文件的大小：**在下载之前，需要获取要下载文件的大小，以便将其分成适当的块。
- 创建多个线程：**创建多个线程来同时下载不同的文件块。可以使用`Thread`类或`Task`类。
- 分配每个线程的下载范围：**将文件大小平均分配给每个线程，确保每个线程下载不同的文件块。
- 下载文件块：**每个线程根据分配的范围下载文件块，然后将其保存到本地。
- 等待所有线程完成：**使用线程同步机制，确保所有线程都完成下载任务。
- 合并文件块：**将下载的文件块按照顺序合并成完整的文件。
- 显示下载进度：**可以使用委托或事件来更新下载进度，确保用户能够看到下载的进展情况。

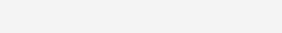
完整实例

以下是一个简单的C#控制台应用程序，用于演示多线程分块下载文件并显示进度。

```
1  using System;
2  using System.IO;
3  using System.Net;
4  using System.Threading;
5
6  class Program
7  {
8      static int numThreads = 4; // 可以根据需要设置线程数
9      static long fileSize;
10     static long blockSize;
11     static long downloadedSize = 0;
12
13     static void Main()
14     {
15         string fileUrl = "https://example.com/largefile.zip";
16         string savePath = "downloadedFile.zip";
17
18         // 获取文件大小
19         fileSize = GetFileSize(fileUrl);
20
21         // 计算每个线程下载的块大小
22         blockSize = fileSize / numThreads;
23
24         // 创建线程数组
25         Thread[] threads = new Thread[numThreads];
26
27         // 下载文件并显示进度
28         for (int i = 0; i < numThreads; i++)
29         {
30             int threadNumber = i;
31             threads[i] = new Thread(() => DownloadFilePart(fileUrl, savePath, threadNumber));
32             threads[i].Start();
33         }
34
35         // 等待所有线程完成
36         foreach (var thread in threads)
37         {
38             thread.Join();
39         }
40
41         Console.WriteLine("下载完成！");
42     }
43
44     static void DownloadFilePart(string fileUrl, string savePath, int threadNumber)
45     {
46         long startByte = threadNumber * blockSize;
47         long endByte = (threadNumber == numThreads - 1) ? fileSize - 1 : startByte + blockSize - 1;
48
49         WebClient client = new WebClient();
50         Stream stream = client.OpenRead(fileUrl);
51
52         // 设置读取的起始位置
53         stream.Seek(startByte, SeekOrigin.Begin);
54
55         // 创建文件流用于保存下载的块
56         using (FileStream fs = new FileStream(savePath, FileMode.OpenOrCreate, FileAccess.Write, FileShare.Write))
57         {
58             byte[] buffer = new byte[1024];
59             int bytesRead;
60
61             while ((bytesRead = stream.Read(buffer, 0, buffer.Length)) > 0)
62             {
63                 fs.Write(buffer, 0, bytesRead);
64                 Interlocked.Add(ref downloadedSize, bytesRead);
65                 DisplayProgress();
66             }
67         }
68
69         stream.Close();
70     }
71
72     static void DisplayProgress()
73     {
74         double progress = (double)downloadedSize / fileSize * 100;
75         Console.WriteLine($"已下载: {progress:F2}%");
76     }
77
78     static long GetFileSize(string fileUrl)
79     {
80         WebRequest request = WebRequest.Create(fileUrl);
81         request.Method = "HEAD";
82
83         using (WebResponse response = request.GetResponse())
84         {
85             long contentLength;
86             if (long.TryParse(response.Headers.Get("Content-Length"), out contentLength))
87             {
88                 return contentLength;
89             }
89             else
90             {
91                 throw new InvalidOperationException("无法获取文件大小。");
92             }
93         }
94     }
95 }
}
```

请注意，此示例使用了`WebClient`和`WebRequest`类来下载文件。在实际应用中，可能需要处理更多的异常情况，并根据需要调整代码。此外，为了简化示例，没有包含对HTTPS、重试机制等的处理。在生产环境中，这些方面需要更多的注意。

相关代码下载地址



重要提示！：取消关注公众号后将无法再启用回复功能，不支持解封！

第一步：微信扫码关注公众号“架构师老卢”

第二步：在公众号聊天框发送`code: 17252`，如： `code: 17252` `获取下载地址`

第三步：恭喜你，快去下载你想要的资源吧