**Chapter 9 异步与并行编程**

—————————————————————————————————————————————

并行编程吓坏了许多开发者，好像这一术语已经成为复杂和难以发现bug的同义词。然而，使用F#中可用的库和语言特性，对并行编程的任何焦虑都将是没有必要的。

事实上，即使并行编程很困难——我认为并不是这样——它仍然值得学习，因为没有它你就不能有效地利用你的硬件潜能。

这一章集中介绍了如何使用F#的异步和并行编程来提高计算速度。到本章结束时，你将能够在不同的环境(线程)中执行代码、使用F#的异步工作流库来精通异步编程，还能利用.NET的并行扩展。在开始之前，让我们定义一些与并行编程相关的概念。

异步编程Asynchronous programming  
 Asynchronous programming describes programs and operations that once started  
are executed in the background and terminate at some “later time.” For example, in  
most email clients, new emails are retrieved asynchronously in the background so  
the user doesn’t have to force checking for new mail.  
Parallel programming  
Parallel programming is dividing up work between processing resources in order  
to speed up execution. For example, converting a song into an MP3 can be  
parallelized—dividing the song into pieces and converting each segment in parallel.  
Reactive programming  
Reactive programming is writing applications in such a way that they respond to  
events as they occur in real time. For example, an application that updates the user  
interface whenever the backing data store has been updated.