Chapter 21. Advanced Threading

We started Chapter 14 with the basics of threading as a precursor to tasks and asynchrony. Specifically, we showed how to start and configure a thread, and covered essential concepts such as thread pooling, blocking, spinning, and synchronization contexts. We also introduced locking and thread safety, and demonstrated the simplest signaling construct, `ManualResetEvent`.

This chapter picks up where Chapter 14 left off on the topic of threading. In the first three sections, we flesh out synchronization, locking, and thread safety in greater detail. We then cover:

- Nonexclusive locking (`Semaphore` and reader/writer locks)
- All signaling constructs (`AutoResetEvent`, `ManualResetEvent`, `CountdownEvent`, and `Barrier`)
- Lazy initialization (`Lazy<T>` and `LazyInitializer`)
- Thread-local storage (`ThreadStaticAttribute`, `ThreadLocal<T>`, and `GetData`/`SetData`)
- Timers

Threading is such a vast topic that we've put additional material online to complete the picture. Visit *http://albahari.com/threading* for a discussion on the following, more arcane, topics:

- `Monitor.Wait` and `Monitor.Pulse` for specialized signaling scenarios
- Nonblocking synchronization techniques for micro-optimization (`Interlocked`, memory barriers, `volatile`)