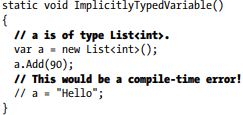
**Dynamic Types and the Dynamic Language Runtime**

NET 4.0 introduced a new keyword to the C# language, specifically, dynamic. This keyword allows you to incorporate scripting-like behaviors into the strongly typed world of type safety, semicolons, and curly brackets. Using this loose typing, you can greatly simplify some complex coding tasks and also gain the ability to interoperate with a number of dynamic languages (such as IronRuby or IronPython), which are .NET savvy. In this chapter, you will be introduced to the C# dynamic keyword and understand how loosely typed calls are mapped to the correct in-memory object using the Dynamic Language Runtime (DLR). After you understand the services provided by the DLR, you will see examples of using dynamic types to streamline how you can perform late-bound method calls (via reflection services) and to easily communicate with legacy COM libraries.

**The Role of the C# dynamic Keyword** Back in Chapter 3, you learned about the var keyword, which allows you to define local variables in such a way that the underlying date type is determined at compile time, based on the initial assignment (recall that this is termed implicit typing). Once this initial assignment has been made, you have a strongly typed variable, and any attempt to assign an incompatible value will result in a compiler error

