Lesson 6: Error Handling

Error Handling Best Practices

Exception handling is the technique of handling runtime errors in your application code. Basically, you have two categories of exceptions: Exceptions that are generated by the application and those that are generated by the runtime. Exceptions should be handled with care -- you should have a good idea of how exceptions should be handled and when they are needed to be handled in your code. In this post, I will present a few tips and best practices for working with exceptions in C#.

The base class for all exceptions in .NET is Exception. All exception classes in the exception hierarchy derive directly or indirectly from this class. The ApplicationException and SystemException classes are derived from the Exception class. The Common Language Runtime (CLR) throws an instance of a type that is derived from SystemException when an error occurs at runtime. Note that you should never catch SystemException or throw an instance of SystemException in your application's code.

When creating custom exception classes, always derive from the Exception class and not from the ApplicationException class. One of the reasons for this is that an instance of ApplicationException is thrown by the application and never by the runtime. In throwing an instance of ApplicationException in your code, you would just increase the call stack without adding much value.