

Lesson 6: Error Handling

Try-catch

An exception is a problem that arises during the execution of a program. A C# exception is a response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero.

Exceptions provide a way to transfer control from one part of a program to another. C# exception handling is built upon four keywords: **try**, **catch**, **finally**, and **throw**.

- **try**: A try block identifies a block of code for which particular exceptions is activated. It is followed by one or more catch blocks.
- **catch**: A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The catch keyword indicates the catching of an exception.
- **finally**: The finally block is used to execute a given set of statements, whether an exception is thrown or not thrown. For example, if you open a file, it must be closed whether an exception is raised or not.
- **throw**: A program throws an exception when a problem shows up. This is done using a throw keyword.

Syntax

Assuming a block raises an exception, a method catches an exception using a combination of the try and catch keywords. A try/catch block is placed around the code that might generate an exception. Code within a try/catch block is referred to as protected code, and the syntax for using try/catch looks like the following:

```
try
{
    // statements causing exception
}
catch( ExceptionName e1 )
{
    // error handling code
}
catch( ExceptionName e2 )
{
    // error handling code
}
catch( ExceptionName eN )
{
    // error handling code
}
finally
{
    // statements to be executed
}
```

You can list down multiple catch statements to catch different type of exceptions in case your try block raises more than one exception in different situations.