Use user-filtered exception handlers

12/14/2020 • 2 minutes to read • +4

In this article

User-filtered exception handlers catch and handle exceptions based on requirements you define for the exception. These handlers use the `catch` statement with the `when` keyword (`Catch` and `When` in Visual Basic).

This technique is useful when a particular exception object corresponds to multiple errors. In this case, the object typically has a property that contains the specific error code associated with the error. You can use the error code property in the expression to select only the particular error you want to handle in that `catch` clause.

The following example illustrates the `catch`/`when` statement.

C# Copy

```
try
{
    //Try statements.
}
catch (Exception ex) when (ex.Message.Contains("404"))
{
    //Catch statements.
}
```

The expression of the user-filtered clause is not restricted in any way. If an exception occurs during execution of the user-filtered expression, that exception is discarded and the filter expression is considered to have evaluated to false. In this case, the common language runtime continues the search for a handler for the current exception.

Combine the specific exception and the user-filtered clauses

A `catch` statement can contain both the specific exception and the user-filtered clauses. The runtime tests the specific exception first. If the specific exception succeeds, the runtime executes the user filter. The generic filter can contain a reference to the variable declared in the class filter. Note that the order of the two filter clauses cannot be reversed.

The following example shows a specific exception in the **catch** statement as well as the user-filtered clause using the **when** keyword.

C# Copy

```csharp
try
{
    //Try statements.
}
catch (System.Net.Http.HttpRequestException ex) when
(ex.Message.Contains("404"))
{
    //Catch statements.
}
```

See also

- Exceptions

Implement a Dispose method

In this article, learn to implement the Dispose method, which releases unmanaged resources used by your code in .NET.

Show more