O – Exception Handling

Code Samples - Documentation

# Examples

The code for real-world programs is typically divided into distinct “layers” in order to make programs more flexible and maintainable. The layer that corresponds to the drivers we have seen so far is called the “Presentation Layer”. It’s called a “presentation layer” because its sole purpose is to interact with a user.

Almost all exception handling should take place in the driver, or presentation layer, of an application. Part of the reason for this is because the presentation layer is the “top” or “front-most” part of the program. When exceptions are handled at this layer, an application can report on the errors that occurred and give opportunity for the user to choose their desired response to the error.

The following examples are used to illustrate this topic.

1. **DemoTestingForExceptions** – This class is a very simple demonstration of how to catch exceptions and how exception handling allows a program to keep running even when a problem occurs.
2. **DemoBookFileAdapter** – This class demonstrates displaying book information read from a file. It also demonstrates attempting to read from a file or directory that does not exist, and capturing an exception thrown by classes within the .Net Framework.
3. (Not Available) **MediaPlayer** – This class demonstrates a complete working program that presents the user with a menu of various tasks related to loading and displaying songs on a CD. This class also demonstrates exception handling.

# DemoTestingForExceptions

This class is a very simple demonstration of how to catch exceptions and how exception handling allows a program to keep running even when a problem occurs.

## Code Solution

public class DemoTestingForExceptions

{

private static ConsoleColor \_Normal = Console.ForegroundColor;

public static void Start()

{

string[] actualAuthors = { "Dan Gilleland", "Nelson McRae", "Nathan Humphrey", "Julie Lutter" };

string actualTitle = "Java: Bean There, Done That";

string actualBarCode = "0-12345678-9";

// Demo a normal creation of a Book object

Console.WriteLine("Testing valid data");

TestBook(actualTitle, actualAuthors, actualBarCode);

// Demo testing the Book class for various combinations of data

Console.WriteLine("Testing empty title");

TestBook("", actualAuthors, actualBarCode);

Console.WriteLine("Testing null title");

TestBook(null, actualAuthors, actualBarCode);

Console.WriteLine("Testing empty author list");

TestBook(actualTitle, new String[0], actualBarCode);

Console.WriteLine("Testing null author list");

TestBook(actualTitle, null, actualBarCode);

Console.WriteLine("Testing empty bar code");

TestBook(actualTitle, actualAuthors, "");

Console.WriteLine("Testing null bar code");

TestBook(actualTitle, actualAuthors, null);

Console.WriteLine("Testing all nulls");

TestBook(null, null, null);

}

private static void TestBook(string title, string[] authors, string barCode)

{

try

{

Book theBook = new Book(title, authors, new ISBN(barCode));

Console.WriteLine("I creatd the following book:");

Console.WriteLine(" Title: " + theBook.Title);

Console.Write(" Authors: ");

foreach (string author in theBook.Authors)

Console.Write(author + ", ");

Console.WriteLine("(" + theBook.Authors.Length + " authors)");

Console.WriteLine(" ISBN: " + theBook.Isbn.BarCode);

Console.WriteLine("-----------------------------------------");

Console.WriteLine();

}

catch (Exception ex)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("!-- " + ex.Message + " --!");

Console.ForegroundColor = \_Normal;

}

}

}

# DemoBookFileAdapter

This class demonstrates displaying book information read from a file. It also demonstrates attempting to read from a file or directory that does not exist, and capturing an exception thrown by classes within the .Net Framework.

public class DemoBookFileAdapter

{

private static ConsoleColor \_Normal = Console.ForegroundColor;

public static void Start()

{

try

{

Book[] myBooks;

myBooks = BookFileAdapter.LoadList("BookList.txt", FileFormat.CSV).ToArray();

DisplayBooks(myBooks);

myBooks = BookFileAdapter.LoadList(@"V:\ThisDriveNameDoesNotExist", FileFormat.CSV).ToArray();

myBooks = BookFileAdapter.LoadList("ThisFileNameDoesNotExist", FileFormat.CSV).ToArray();

DisplayBooks(myBooks);

}

catch (DirectoryNotFoundException ex)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("!-- " + ex.Message + " --!");

Console.ForegroundColor = \_Normal;

}

catch (FileNotFoundException ex)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("!-- " + ex.Message + " --!");

Console.ForegroundColor = \_Normal;

}

catch (Exception ex)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("!-- " + ex.Message + " --!");

Console.ForegroundColor = \_Normal;

}

}

private static void DisplayBooks(Book[] myBooks)

{

for (int index = 0; index < myBooks.Length; index++)

{

Console.WriteLine("Title : " + myBooks[index].Title);

Console.WriteLine("ISBN : " + myBooks[index].Isbn.BarCode);

Console.WriteLine("Authors: ");

foreach (string author in myBooks[index].Authors)

Console.WriteLine("\t" + author);

Console.WriteLine();

}

}

}