

Practica Excepciones

Try-catch

Emiliano Montesdeoca del Puerto | 1DAWB | 07-11-2016

# Índice

1. Enunciado
2. Objetivo
3. Código
4. Código en funcionamiento
5. Tratamiento de excepciones

# Enunciado

1. Tenemos el programa diademanana.cs
2. Añadir el tratamiento de excepciones a este código.
3. El nuevo código con el tratamiento de excepciones.

# Objetivo

En esta práctica, el objetivo es añadir excepciones al código y comprobar que funcionan.

De la página de MSDN de Microsoft, las características de excepciones:

Las características de control de excepciones del lenguaje C# proporcionan una manera de afrontar cualquier situación inesperada o excepcional que se presente mientras se ejecuta un programa. El control de excepciones utiliza las palabras clave try, catch y finally para intentar realizar acciones que podrían plantear problemas, controlar errores cuando considere que sea razonable y limpiar los recursos después. Pueden generar excepciones Common Language Runtime (CLR), .NET Framework, las bibliotecas de otros fabricantes o el código de aplicación. Las excepciones se crean mediante la palabra clave throw.

# Codigo

static void Main(string[] args)

{

int dd, mm, aa;

int dds, mms, aas;

int dias = 0;

Console.Write("Introduzca Día: ");

dd = ExceptionInt();

Console.Write("Introduzca Mes: ");

mm = ExceptionInt();

Console.Write("Introduzca Año: ");

aa = ExceptionInt();

if (dd >= 0 && dd <= 31 && mm >= 0 && mm <= 12 && aa >= 0)

{

dds = dd + 1;

mms = mm;

aas = aa;

switch (mm)

{

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

case 12:

dias = 31;

break;

case 2:

if ((aa % 4 == 0) && (aa % 100 != 0) || (aa % 400 == 0))

dias = 29;

else

dias = 28;

break;

case 4:

case 6:

case 9:

case 11:

dias = 30;

break;

}

if (dds > dias)

{

dds = 1;

if (mms == 12)

{

mms = 1;

aas++;

}

else

mms++;

}

Console.WriteLine("\n\n\n");

Console.Write("Fecha: ");

Console.Write("{0}//{1}//{2}", dds, mms, aas);

Console.ReadLine();

}

else

{

Console.WriteLine("Fecha erronea.");

Console.ReadLine();

}

static int ExceptionInt()

{

int res = 0;

bool success = false;

do

{

try

{

res = Convert.ToInt32(Console.ReadLine());

success = true;

}

catch (Exception)

{

Console.Write("Valor erroneo, introduzca de nuevo: ");

}

} while (!success);

return res;

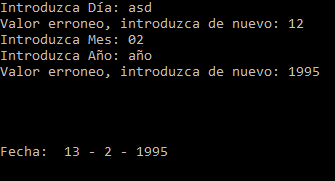
}

# Código en funcionamiento

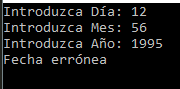
La forma en el que la función **ExceptionInt** funciona es preguntando un valor y comprobando si ese valor introducido es un numero o no.



El código nunca sigue hasta que se ha introducido el código bien:



También se ha modificado la función comprueba, se hace el cálculo del día siguiente si todos los valores están dentro de unos rangos aceptados.



# Tratamiento de excepciones

La forma en la que se ha hecho la función para el tratamiento de excepciones es simple: si a la hora de introducir un número, introduces una cadena de caracteres, esta no es válida y se tiene que volver a introducir.

static int ExceptionInt()

{

int res = 0;

bool success = false;

do

{

try

{

res = Convert.ToInt32(Console.ReadLine());

success = true;

}

catch (Exception)

{

Console.Write("Valor erroneo, introduzca de nuevo: ");

}

} while (!success);

return res;

}

La forma en la que esta funciona trabaja es un **do-while**, que mientras no se ponga a verdadero una variable seguirá repitiendo.

Esa variable en cuestión es un booleano llamado **success** que se ha instanciado falsa desde el principio y siempre será falsa hasta que no se dé un valor que cumpla con los requisitos.

Dentro del try, tenemos el Console.ReadLine que leerá lo que introducimos por teclado, y ahí, si introducimos una cadena de texto saltara una excepción, que nos mostrará un mensaje y se volver a repetir el bucle.

Hasta que no se introduce un valor correcto, no pondrá a verdadero la variable que hará que el bucle siga repitiéndose.

En cuanto a la comprobación de si la fecha introducida es correctase ha hecho una modificación del código, que comprueba antes de hacer un calculo

if (dd >= 0 && dd <= 31 && mm >= 0 && mm <= 12 && aa >= 0)

{

dds = dd + 1;

mms = mm;

aas = aa;

switch (mm)

{

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

case 12:

dias = 31;

break;

case 2:

if ((aa % 4 == 0) && (aa % 100 != 0) || (aa % 400 == 0))

dias = 29;

else

dias = 28;

break;

case 4:

case 6:

case 9:

case 11:

dias = 30;

break;

}

if (dds > dias)

{

dds = 1;

if (mms == 12)

{

mms = 1;

aas++;

}

else

mms++;

}

Console.WriteLine("\n\n\n");

Console.Write("Fecha: ");

Console.Write("{0}//{1}//{2}", dds, mms, aas);

Console.ReadLine();

}

Este **if** se encarga de leer todos los números introducidos y comprarlos con unos requisitos.

1. Si el día está entre 0 y 31.
2. Si el mes esta entre 0 y 12.
3. Y si el año es mayor a 0(en este caso).

Entonces, si el numero cumple esos requisitos, el código ejecuta la parte en la que calcula el día siguiente y lo muestra.

Si no cumple esos requisitos, te muestra que la fecha es errónea.