Exceptions:

Se lanzan cuando ocurre un error en tiempo de ejecucion

Derivan directa o indirectamente de la clase Exception

Si se lanza una exception y no es atrapada en un bloque catch el programa deja de funcionar.

Posee propiedad Message la cual devuelve el msj de la exception.

Detienen el flujo del programa, si no se hace nada deja de funcionar.

Generan msjs de error significativos

Donde debe ir la exception mas potente??

El ultimo bloque catch debe ser el general.

Ningun bloque catch despues de un try captura la exception lanzada en ese try:

El programa deja de funcionar

El runtime desenreda la pila de llamadas en busca de un catch que capture la exception

Finally

Se ejecuta siempre.

Test Unitarios:

Utilizamos los metodos estaticos de Assert comprueba si la accion del metodo se comporta como es esperado

Creamos una instancia de la clase assert en nuestros metodos de test para llevar a cabo nuestras pruebas

Las clases destinadas a la ejecucion de prueba deben tener la etiqueta [TestClass] y sus metodos de prueba la etiqueta [TestMethod]

Escribir casos de prueba para cada funcion no trivial o metodo en el modulo de forma independiente

Patron AAA (arrange, act, assert)

Assert:

Puede controlarse desde fuera del metodo a traves de la etiqueta enviando un except

Por etiqueta a traves del timeout

Pruebas de integracion o pruebas integrales:

Asegurar el correcto funcionamiento del sistema

Ambito del desarrollo de software, despues de pruebas unitarias.

Prueba que todo el grupo funcione bien

Prueba funcional

Basada en la ejecicion, revision y retroalimentacion de las funcionalidades previamente diseñadas

Se hacen mediante diseño de modelos de prueba. Evaluan cada opcion del paquete informatico

Como se crean los test unitarios:

Creado una clase de test

Creando un proyecto de test

Que clase explicita el resultado del test

Assert

Pruebas que indican que todos los elementos funcionan correctamente en conjunto:

Pruebas integrales

Archivo de Texto:

StreamWriter

Streamreader

Primero se inicia el new stream, luego se hace readLine o writeLine

Debe cerrar el archivo sw.Close() Tambien puede usar using

Ecoding e - Tipo de codificacion

File.Exist - si la ruta es valida y si puede leer el archivo

.Copy - copia el archivo

.Delete - elimina

GetFiles() - devuelve el nombre del archivo y ruta especifica

SpecialFolders

Enviroment.getFolderPath(enviroment.SpecialFolder.Desktop)

Serializacion:

Proceso de convertir un objeto en una secuencia lineal de bytes

Se reconstruye mediante Deserealizacion, proceso inverso.

XML:

Debe tener constructor por defecto

No se puede serializar metodos

Propiedades y atributos publicos

No lo hace si es solo lectura

xmlTextReader: manera de leer archivos XML

XMLSerializer:

Serialize: serializa el objeto y escribe doc XML usando el stream especifico

Deserialize: deserealiza el doc XML por el stream

Binario:

Todos los atributos y propiedades, publicos o privados

La clase lleva la etiqueta [Serializable]

BinaryFormatter: serializa y deserializa

La clase debe tener constructor por defecto

Base de Datos:

Comand: procedimiento de almacenado o una instruccion de SQL server

SqlConnection(connectionstring)

command.Comandtype = commandtype.text

Comando.connection = conexion

Comando.comandtect = “select...”

sqlDataReader x = comando.executeReader()

Insert, update, delete = excecuteNonQuery()

Para que se usa un Connection String en una conexion a base de datos?

Datos de usuario, servidor, seguridad, nombre de base de datos

Threads:

Una secuencia de tareas encadenadas

Tarea que puede ser ejecutada a la vez que otra

Comparten recursos, en conjuntos son conocidos como procesos

Cuando finaliza libera recursos

Puede ser parametrizado

Invoke

begin Invoke para form

Cuando invocamos Sleep de Thread dentro de un metodo invocado por un thread secundario:

Se duerme al hilo secundario durante la cantidad de milisegundos especificada

Eventos:

Es un modo que tiene una clase de proporcionar notificaciones a sus clientes cuando ocurre algo

Es un msj enviado por un obj para indicar una accion invocada

Tiene un emisor que produce el evento y un receptor que lo captura

Receptor o manejador.

Delegado enlaza al emisor con el receptor

Son subprocesos que puede disparar una clase y que corren en paralelo al hilo de ejecucion principal

Se los declara mediante delegados

Es un tipo que representa referencias a metodos con una lista de parametros determinada y un valor devuelto

Los manejadores son funciones asociadas a un evento que se ejecutan cuando se dispara el mismo

Un evento solo puede tener un manejador asociado

Declarar delegado:

Public delegate void MiDelegado(object sender, EventArgs e)

Declarar evento:

Public event MiDelegado ElEvento;

Invocar el evento:

Public void MetodoManejador()

Asociar el evento a un manejador

objEmisor.evento += MetodoManejador()

Metodo de Extension:

Se define como metodos estaticos pero se los llama usando sintaxis de instancia

Permiten agregar metodos a los tipos ya existentes sin recompilar o modificar

Son una clase especial de un metodo estatico pero se los llama como si fueran metodos de instancia

Es valido para clases como interfaces

Metodo(this string dato)

Generics:

Pueden utilizarse multiples generics para cada clase, metodo, etc. donde se ejecute

Al instanciar el objeto, reemplazare el “comodin” por un tipo de dato especifico

Los metodos pueden recibir tipos genericos pero no retornarlos

Son clases parametrizadas que pueden contener lo mismo que una clase comun

Son un tipo de clase abstracta que puede recibir parametros

No es necesario que la clase tambien sea generic

Que hace la palabra Where en un metodo parametrizado

Restringe el parametro que puede utilizarse

Restricciones

Todos los tipos de valor deben tener un constructor sin parametros accesible

Struct implica la restriccion new() y no se puede combinar con struct

Unmanaged implica la restriccion struct. No se puede combinar con struct o new()

Interfaces:

Class MiClase : IMiInterface

Contrato que establece una clase en la que se asegura que implementara un conjunto de metodos

Una interfaz contiene miembros sin implementacion los cuales deben ser obligatoriamente implementados en clases que implementen interfaz.

No permite especificar atributos

Una clase puede implementar mas de una interfaz si se requiere

Aunque no se especifique, por definicion todos los miembros de una interfaz son publicos y abstractos

Formas de instanciarlas junto con herencias

Propiedades sin implementacion

Parecido a una herencia multiple

Explicita : IInterface.Metodo

Que es un hilo?

Un proceso o rutina

Una serie de metodos ordenados

Una secuencia de tareas encadenadas

Ninguna

Cuando se termina un hilo:

Cuando se ejecuta el metodo abort

Cuando termina de ejecutar el metodo

Cuando finaliza el hilo padre

Cuando ocurre un error

Los delegados

Son metodos ejecutables

Son colecciones de metodos

Son una lista de punteros a metodos

Son un puntero a funcion

Si tengo una tabla con id y descripcion, que es correcto?

Select \* from Productos

Insert Productos (id, descripcion) values (‘Producto’)

Insert Productos (id, descripcion) values (4, ‘Producto)

Insert Productos (descripcion) values (‘Producto’)

Al ejecutar Thread:

No finaliza si se culmina la ejecucion del hilo principal

Este finaliza y libera el bloque de memoria al culminar su ejecucion normal

Podremos ejecutar el metodo Abort mas de una vez para el mismo metodo sin error

Proporcionaremos un medio apropiado para que los objetos puedan señalizar cambios de estado que pueden resultar utiles para los clientes de ese objeto}

Se creara un subproceso que tomara recursos de maquina dentro de un proceso que puede ser mas grande y contar con otros subprocesos