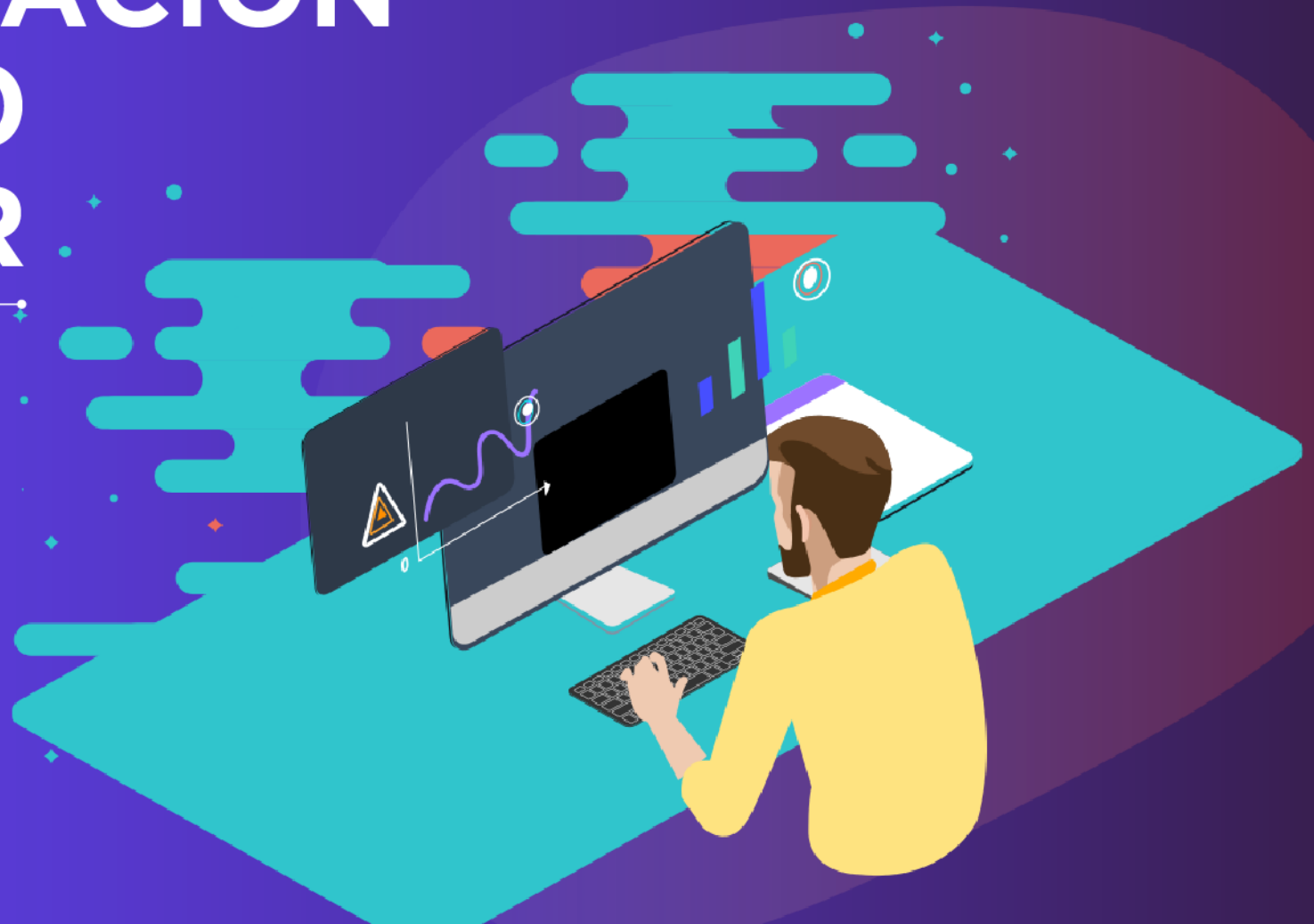
PROGRAMA

.NET ESPECIALIZACIÓN: NET

ESPECIALIZACIÓN ASP.NET 5.0 DEVELOPER



CURSO VIRTUAL

Fundamentos de Programación

Repaso general

Instructor: Erick Aróstegui
earostegui@galaxy.edu.pe

GALAXY TRAINING

TEMAS

01	Repaso General

Instalación de Visual Studio

Paso1:Vamos a comprobar que nuestra máquina está lista para que instalemos Visual Studio 2019. Tendremos que realizar algunas actualizaciones y liberar algo de espacio.
Paso2: Ahora, vamos a obtener nuestra descarga de Visual Studio 2019 para prepararnos para la instalación.
Paso3 :Ejecute el programa de instalación y elija instalar los componentes de desarrollo móvil. Asegúrese de que no se estén ejecutando otras versiones de Visual Studio cuando instale Visual Studio 2019.
Paso4:Ya está listo para la acción. Ya puede explorar o disfrutar de la ventaja que le Visual Studio

Variables

Tipo de Variable	Descripción
Int32	Entero con signo
Int64	Entero con signo
Doublé	Tipo de punto flotante con precisión
Char	Un carácter Unicode
Boolean	Tipo boolean lógico
String	Una secuencia de caracteres

Estructura Condicional

nos permiten ejecutar una serie de instrucciones si cumple una determinada condición que nosotros le indiquemos.

- if: le indicamos una condición si esta es verdadera se ejecuta, sino no se ejecuta las instrucciones de dentro.
- if – else: es como el anterior solo que después de cerrarse la llave de if, se añade else sin indicarle ninguna condición.
- if -elseif: esta estructura es como una mezcla de los anteriores, esto nos permite, que si no se cumple la condición podamos indicar otra condición para hacerlo aún mas especifico.
- Switch: esta estructura condicional de selección múltiple, le damos un valor y una lista de casos y si cumple alguno de los casos ejecuta las instrucciones asociadas a ella

Estructura Repetitiva

Se refieren a procesos repetitivos dentro de con contexto hasta que cumpla con la condición, termina el proceso una vez cumplida la condición

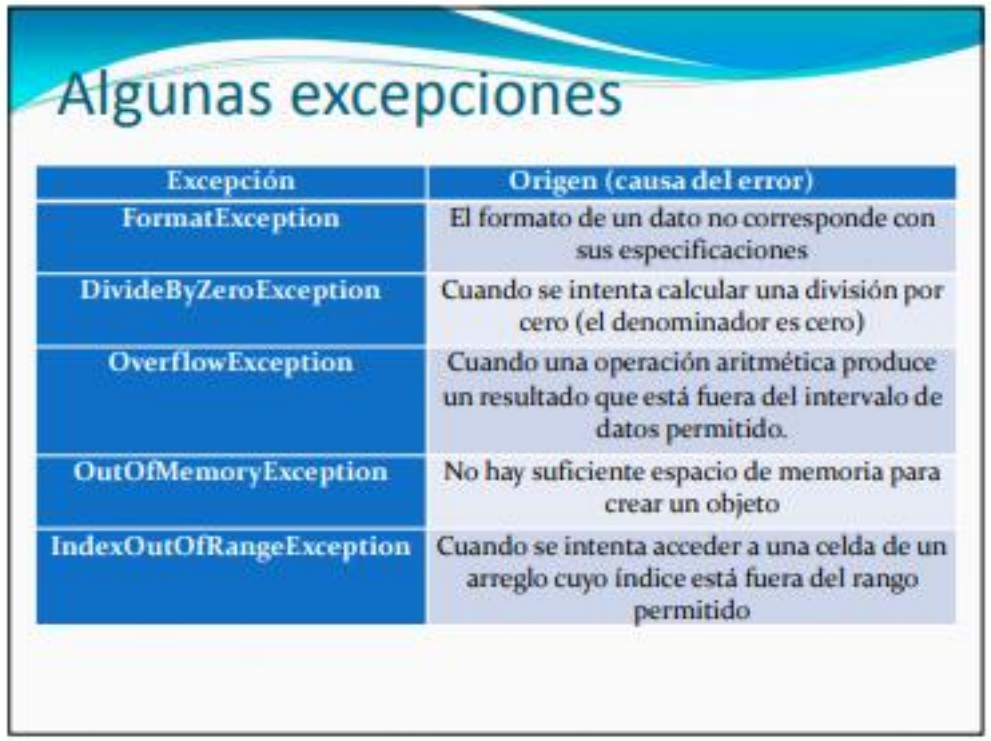
WHILE: repite todo un proceso hasta que cumpla la condición indicada.

DO WHILE:es igual que la anterior, la diferencia es que la primo hace un proceso luego viene la condición,

FOR: esta estructura se caracteriza por tener un punto de inicialización, hasta un punto determinado y un incremento, es decir de un punto a otro punto

Control de excepciones

Las excepciones ocurren cuando no lo esperamos, por eso cuando suceden es importante capturar toda la información posible y guardarla en un Log .En general con tres de las propiedades de la excepción suele haber suficiente información como para saber la causa del error y poder corregirlo.



Manejo de excepciones en c#

C# envía una excepción cuando ocurre un error en el programa y detiene su ejecución. Si se desea que la aplicación siga ejecutándose después del error, entonces se usa:

- try para poner en alerta al programa sobre el código que puede lanzar una excepción.
- catch para capturar y manejar cada excepción que se lance.
- finally código que se ejecutará haya o no excepciones.

Cuando el manejador termina se pueden hacer dos cosas:

- Reanudar la ejecución del bloque
- Terminar la ejecución del bloque y devolver el control al punto de invocación.

Arreglos

Los arreglos en C# son indexados iniciando en cero (0). La forma de trabajo es muy similar a la mayoría de lenguajes pero hay lagunas diferencias que notarán.

- **Arreglo de tipo Vector** podemos indicar el tamaño del arreglo luego de la declaración. Esto nos permite decidir que tamaño va a tener el arreglo según lo que necesitemos.
- **Arreglos Multi-dimensionales** las dimensiones se manejan por medio de un par de corchetes, dentro de los que se escriben los valores de cada dimensión, separados por comas.

Colecciones

proporcionan una forma más flexible de trabajar con grupos de objetos. puede asignar una clave a cualquier objeto que coloque en la colección para poder recuperar rápidamente el objeto usando la clave. Las colecciones fundamentales que podemos hacer uso en nuestros proyectos son:

- Queue<T> : Implementa el concepto de una cola (FIFO - Fist In First Out - Primero en entrar primero en salir)
- Stack<T> : Implementa el concepto de una pila (LIFO - Last In First Out - Ultimo en entrar primero en salir)
- List<T> : Administra una lista de elementos accesible mediante un subíndice como los vectores pero con la ventaja de que puede crecer en forma dinámica.
- LinkedList<T> : Administra una lista doblemente encadenada con las facilidades de insertar y extraer elementos en cualquier parte de la lista en forma muy eficiente.
- Dictionary<TKey, TValue> Facilita administrar una lista de valores accesibles mediante una clave

Queue & Stack, tuple & ValueTuple

Queue y Stack son objetos de colección en el espacio de nombres System.Collection. La clase Queue rastrea los objetos en base al primero en entrar y al primero en salir, mientras que la clase Stack rastrea los objetos en el primero en entrar y el último en salir.

Los constructores de Tuple nos permite crear tuplas de hasta 8 tipos de datos. Como clase genérica, podemos crear una tupla del tipo de datos que nos parezca oportuno.

ValueTuple es una estructura que refleja un objeto Tuple. Además, posee métodos estáticos que nos permite crear tuplas por valor. Otra característica de ValueTuple es que es mutable a diferencia de Tuple que es inmutable tal y como veíamos más arriba.

La gran diferencia entre Tuple y ValueTuple es que System.ValueTuple es una estructura y un tipo por valor, mientras que System.Tuple es un tipo por referencia como las clases.

Leer y escribir un archivo

El ReadLine método lee cada línea de texto e incrementa el puntero de archivo a la siguiente línea al leer. Cuando el ReadLine método alcanza el final del archivo, devuelve una referencia nula.

El siguiente código usa la StreamWriter clase para abrir, escribir y cerrar el archivo de texto. El StreamWriter método escribe una línea completa de texto en el archivo de texto.

POO

Un tipo **struct** es una construcción de programación utilizado para definir tipos personalizados. Los tipos struct se usan para encapsular pequeños grupos de variables relacionadas y representadas como un solo elemento. Para crear una estructura utilizamos la palabra mágica struct, las estructuras pueden hacer lo mismo que una clase, pero funcionan de manera diferente

Las clases en C# se definen de forma parecida a los registros (struct), sólo que ahora, además de variables(atributos), también incluirán funciones (metodos) Atributos y métodos formarán parte de "un todo", en vez de estar separados en distintas partes del programa

Los **atributos** proporcionan una manera de asociar la información con el código de manera declarativa. También pueden proporcionar un elemento reutilizable que se puede aplicar a diversos destinos.

Los **constructores** permiten al programador establecer valores predeterminados, limitar la creación de instancias y escribir código flexible y fácil de leer. podemos definir un método que se ejecute inicialmente y en forma automática. Este método se lo llama constructor.

Un **método** es un bloque de código que contiene una serie de instrucciones. Un programa hace que se ejecuten las instrucciones al llamar al método y especificando los argumentos de método necesarios.

Un **delegado** es un tipo que representa referencias a métodos con una lista de parámetros determinada y un tipo de valor devuelto. Cuando se crea una instancia de un delegado, puede asociar su instancia a cualquier método mediante una signatura compatible y un tipo de valor devuelto.

Herencia y Sobrecarga de métodos

La herencia es una característica de los lenguajes de programación orientados a objetos que le permite definir una clase base que proporciona una funcionalidad específica y definir clases derivadas que heredan o anulan esa funcionalidad.

La sobrecarga de métodos es la creación de varios métodos con el mismo nombre, pero con diferentes firmas y definiciones. Se utiliza el número y tipo de argumentos para seleccionar qué definición de método ejecutar, permite definir dos o más métodos con el mismo nombre, pero que difieren en cantidad o tipo de parámetros.
Esta característica del lenguaje nos facilita la implementación de algoritmos que cumplen la misma función pero que difieren en los parámetros.

Interfaces

Las **interfaces** son una abstracción estupenda que nos ofrecen la mayor parte de los lenguajes de programación orientados a objetos. Básicamente nos permiten definir un "contrato" sobre el que podemos estar seguros de que, las clases que las implementen, lo van a cumplir.

Asociaciones de clases

En una relación de uno a uno, un registro de una tabla se asocia a uno y solo un registro de otra tabla.

En una relación de uno a muchos, un registro de una tabla se puede asociar a uno o varios registros de otra tabla.

Una relación de muchos a muchos se produce cuando varios registros de una tabla se asocian a varios registros de otra tabla.

Programación Funcional

la programación funcional o functional programming se centra en las funciones. En un programa funcional. Todos los elementos pueden entenderse como funciones, La programación funcional ofrece un alto grado de abstracción, ya que está basada en el concepto matemático y el principio de función

Expresión Lambda

son argumentos que se pasan a funciones de orden superior, o se usan para construir el resultado de una función de orden superior que necesita devolver una función.1 Si la función solo se usa una vez o un número limitado de veces, una expresión lambda puede ser sintácticamente más simple que usar una función con nombre.
Las funciones lambda son muy comunes en los lenguajes de programación funcional y en otros lenguajes con funciones de primera clase, en los que cumplen el mismo papel para el tipo de función que los literales para otros tipo de datos t.

Delegados

Func proporciona un soporte para funciones anónimas parametrizadas. Los tipos principales son las entradas y el último tipo es siempre el valor de retorno.

Action son como métodos vacíos, por lo que solo tienen un tipo de entrada. No se coloca ningún resultado en la pila de evaluación.

Inmutabilidad y Colecciones

Inmutabilidad es un objeto cuyo estado no puede ser modificado una vez creado.1 Es el opuesto a los «objetos mutables», que pueden ser modificados tras su creación. En algunos casos un objeto puede ser considerado como inmutable aunque algunos de sus atributos internos cambie, siempre y cuando el estado del objeto parezca no cambiar desde un punto de vista externo al mismo.

Las colecciones de clases de C# son un conjunto de clases diseñadas específicamente para agrupar objetos y llevar a cabo tareas con ellos. Las colecciones proporcionan una forma más flexible de trabajar con grupos de objetos. A diferencia de las matrices, el grupo de objetos con los que trabaja puede crecer y reducirse dinámicamente a medida que cambian las necesidades de la aplicación.

Funciones de orden superior, Tuplas

funciones de orden superior son funciones que cumplen al menos una de las siguientes condiciones:

- Tomar una o más funciones como entrada
- Devolver una función como salida

Una Tupla es una estructura de datos que nos permite almacenar hasta 8 valores diferentes, de diferentes tipos, que están relacionados de algún modo y usando una sola variable.

Cierres y Recurrencias

Los **cierres** se producen inmediatamente para que se cierre una aplicación y no se puede cancelar(Cierra manualmente un objeto Window.).

La **recurrencia** hace referencia a una técnica definición de conceptos en que el concepto definitivo es usado en la propia definición

Función asíncrona y parcial

La **función asíncrona** evita que nuestras tareas a realizar pasen por el mismo hilo ejecutando un nuevo hilo cuando esta es llamada de esta manera no se quedará en espera a que la tarea anterior termine, sino, comenzará la tarea usando un nuevo hilo.

La función parcial es una relación que asocia elementos de un conjunto con, como máximo, uno de los elementos de otro conjunto ,llamado codominio. En cualquier caso, no es necesario que todos los elementos del dominio estén asociados con algún elemento del codominio.

ADO.NET y Linq

ADO.NET es un conjunto de clases que exponen servicios de acceso a datos para programadores de .NET Framework. ADO.NET ofrece abundancia de componentes para la creación de aplicaciones de uso compartido de datos distribuidas.

LINQ permite a los desarrolladores realizar consultas basadas en conjuntos en el código de su aplicación, sin tener que utilizar un lenguaje de consulta independiente. Puede escribir consultas LINQ en varias fuentes de datos enumerables como estructuras de datos en memoria, documentos XML, bases de datos SQL y DataSetObjetos.

Creación de base de datos

Una BD es una estructura de datos que organiza los datos en columnas y filas; cada columna es un campo (o atributo) y cada fila un registro, cada campo debe tener su respectivo tipo de dato en base a los datos que se le registrarán.

CREATE DATABASE : este comando se utiliza para indicar que una base de datos se esta creando seguidamente especificamos el nombre de la base de datos .

CREATE TABLE: este comando se utiliza para indicar que una tabla se va a crear, seguidamente especificamos el nombre de la tabla.

DDL, DML

Los comandos **DDL** (Data Definition Language) es un lenguaje que permite a los programadores de la misma llevar a cabo las tareas de definición de las estructuras que almacenarán los datos así como de los procedimientos o funciones que permitan consultarlos.

Los comandos **DML** (Data Manipulation Language) es un idioma proporcionado por los sistemas gestores de bases de datos que permite a los usuarios de la misma llevar a cabo las tareas de consulta o modificación de los datos contenidos en las Bases de Datos del Sistema Gestor de Bases de Datos.

PREPÁRATE
PARA SER EL
MEJOR

+ ENTREMIENTO
EXPERIENCIA

BIENVENIDOS.

GRACIAS
POR TU PARTICIPACIÓN

Por favor, bríndanos tus comentarios
y sugerencias para mejorar nuestros servicios.

GALAXY
TRAINING


GALAXY
TRAINING