UAI Universidad Abierta Interamericana

01 – PROGRAMACIÓN I

MG. NICOLÁS BATTAGLIA

Ti UAI TECNOLOGIA INFORMATICA

UAIOnline ultra

UAI Universidad Abierta Interamericana

UNIDAD I

INTRODUCCIÓN A LA PROGRAMACIÓN VISUAL - CLASE 2

Ti UAI TECNOLOGIA INFORMATICA

UAIOnline ultra

TEMAS

- CTS Sistemas común de tipos.
- Tipos por valor y por referencia.
- Boxing y Unboxing.
- El tipo Object.
- El tipo String.
- El tipo Date.
- Tipos Numéricos.
- Conversión de tipos.
- Generación de números aleatorios.
- Introducción a los Formularios.
- Formularios MDI.
- Menues.

UAIOnline ultra

UAIOnline ultra

CTS SISTEMAS COMÚN DE TIPOS

- Sistema común de tipos. Define como se declaran, utilizan y administran los tipos en el Common Languaje Runtime (CLR). Es un soporte importante para la integración de los lenguajes.
- Funciones
  - Establece un marco de ayuda para integrar varios lenguajes
  - Define reglas que los lenguajes deben seguir para asegurar que los objetos escritos en varios lenguajes puedan interactuar
  - Proporciona una biblioteca que contiene tipos de datos primitivos (bool, int, char, etc)
- Define cómo funcionan los tipos en el Common Language Runtime

UAIOnline ultra

TIPOS POR VALOR Y REFERENCIA

- Define un conjunto común de “tipos” de datos
- Todo lenguaje de programación debe implementar los tipos definidos por el CTS
  - Todo tipo hereda directa o indirectamente del tipo OBJECT
  - Tipos de VALOR (Stack)y de REFERENCIA (Heap

UAIOnline ultra

TIPOS POR VALOR Y REFERENCIA

Tipos de Datos implementados según su clasificación. Esta clasificación es acorde a si un variable almacena su valor o es un puntero (referencia) a los datos.

- Por Valor
  - Mantiene los datos dentro de su propia asignación de memoria
  - Todos los tipos numéricos, boolean, char, Date
  - Se crean en el STACK de la memoria
- Por Referencia
  - Mantiene una referencia a la posición de memoria donde están los datos
  - Object, String, Arrays, Tipos Class
  - Se crean en el "HEAP (monton)" de la memoria

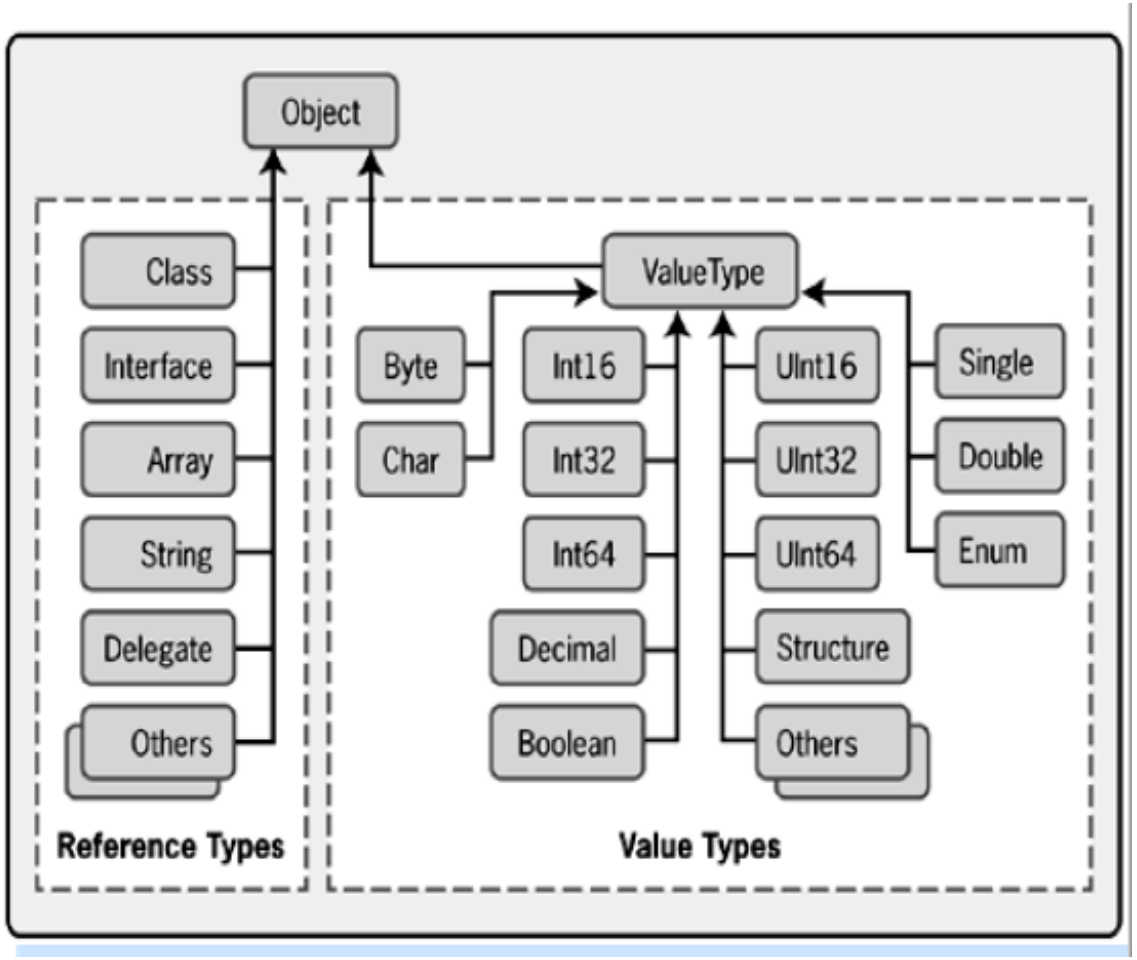
UAIOnline ultra

TIPOS POR VALOR Y REFERENCIA

- Gestion de memoria en C# se hace automáticamente
- Cuando .NET ejecuta un programa guarda los elementos en dos tipos diferentes de memoria: stack y heap.
- el Stack es responsable por hacer el seguimiento de la ejecución de lo que está en nuestro código.
- El Heap es responsable por hacer seguimiento de la mayor parte de nuestros datos
- El Stack lo podemos imaginar como un conjunto de cajas ordenadas una encima de otra. Cada vez que llamamos a un método, para hacer el seguimiento de lo que ocurre en la aplicación colocamos una caja en la parte superior del Stack. Una vez que el método es ejecutado retiramos esta caja y la tiramos a la basura.
- El Heap es similar, solo que su función es mantener la información. Lo que existe en el Heap puede ser accedido en cualquier momento sin limitaciones en lo que se puede acceder como ocurre en el Stack.

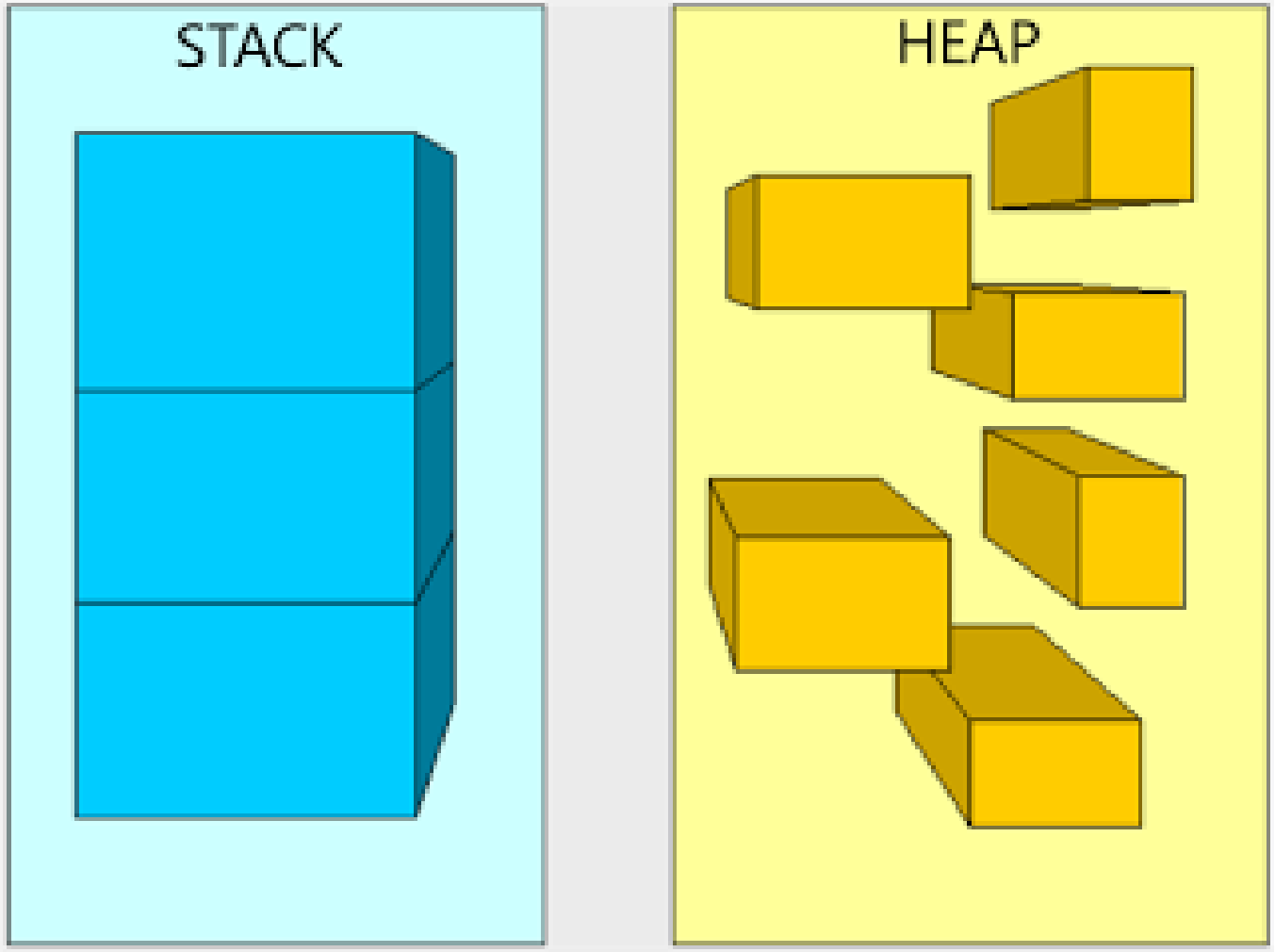
UAIOnline ultra

TIPOS POR VALOR Y REFERENCIA



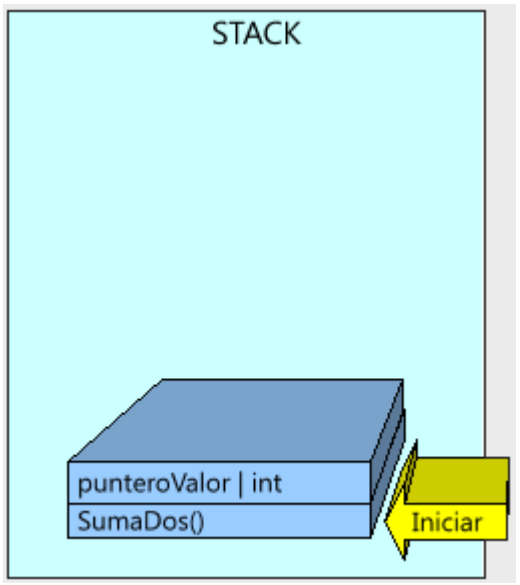
UAIOnline ultra

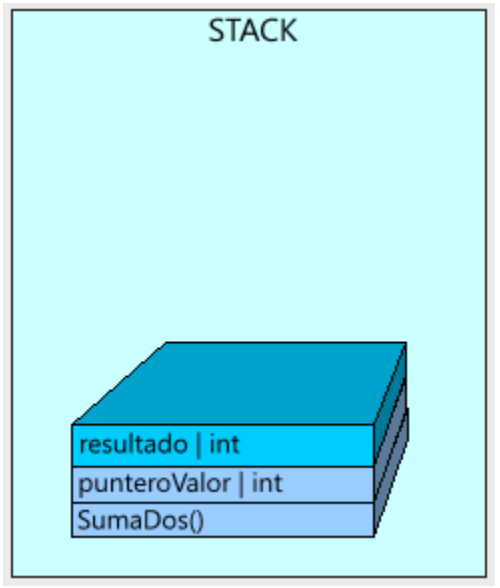
TIPOS POR VALOR Y REFERENCIA

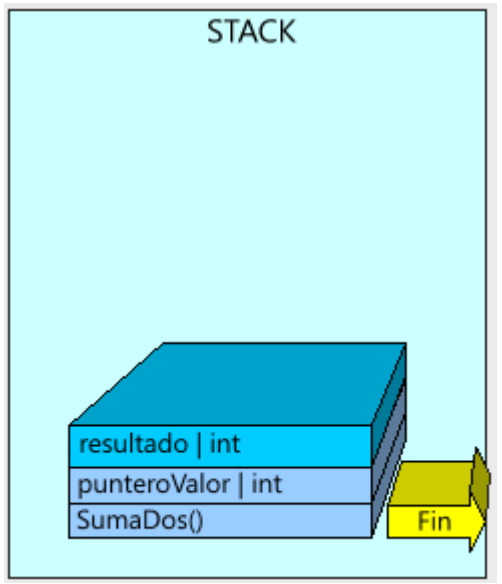


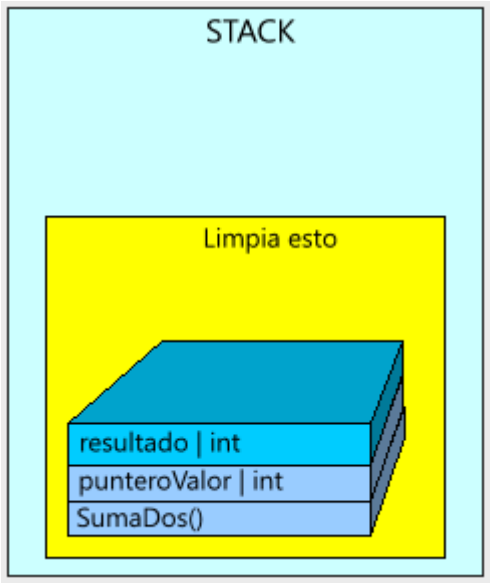
TIPOS POR VALOR Y REFERENCIA

- **public int SumaDos(int valor)**
  {
  int resultado;
  resultado = valor + 2;
  return resultado;
  }



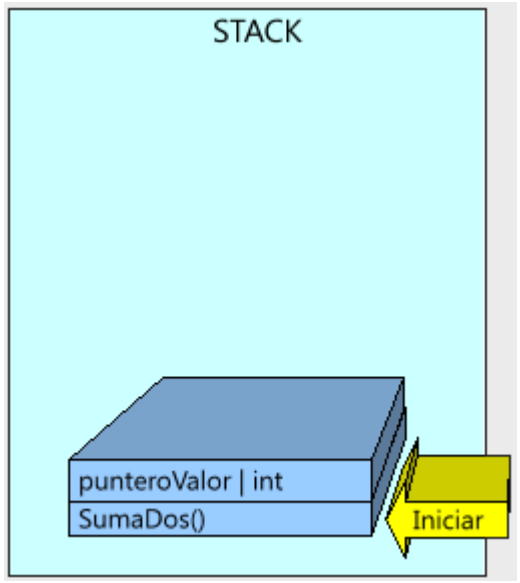


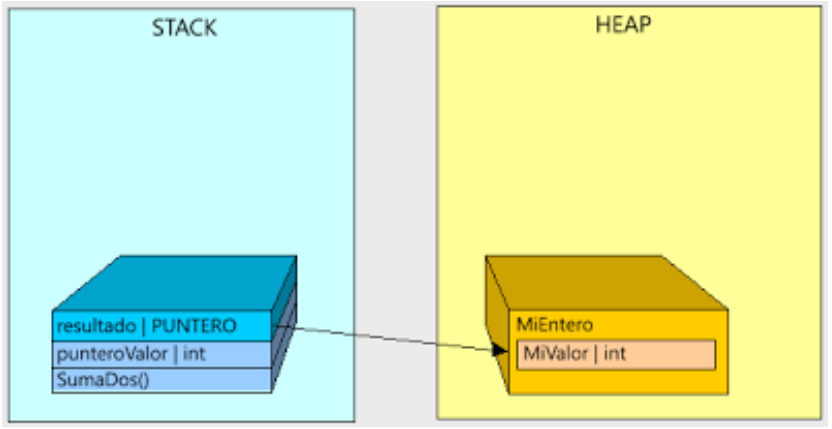


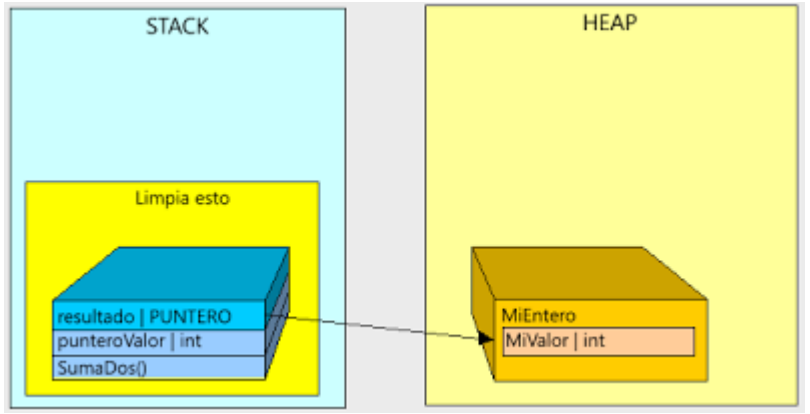


TIPOS POR VALOR Y REFERENCIA

- public MiEntero SumaDos(int punteroValor)
  {
  MiEntero resultado = new MiEntero();
  resultado.MiValor = punteroValor + 2;
  return resultado;
  }

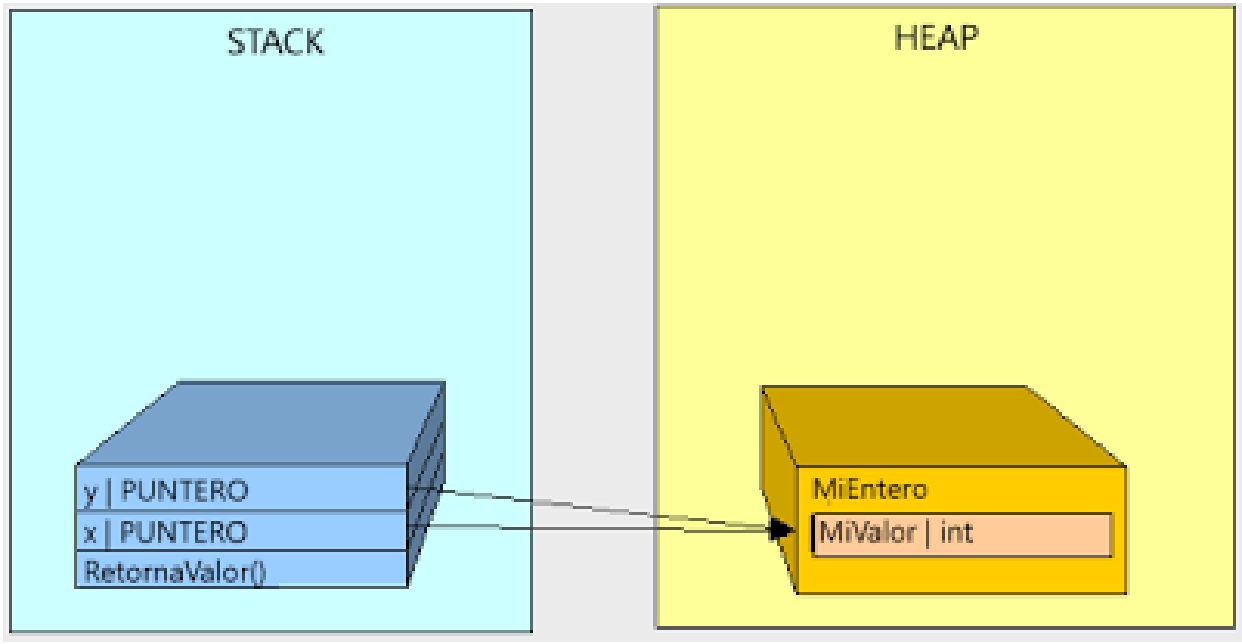






TIPOS POR VALOR Y REFERENCIA

```
public int RetornaValor()
{
        MiEntero x;
        x.MiValor = 3;
        MiEntero y;
        y = x;
        y.MiValor = 4;
        return x.MiValor;
}
```



BOXING Y UNBOXING

UAIOnline ultra

- Boxing
  - Convertir un valor a un tipo por valor (int, double, float, Guid, etc) a un tipo por referencia (System.Object, System.String)

En la siguiente línea la variable i se asigna a un tipo object (boxing)

Ejemplo

```
Int i = 123;
Object o = i;       '
```

BOXING Y UNBOXING

UAIOnline ultra

- Unboxing
  - Convertir un tipo por referencia a un tipo por valor

Object o=123;

Int i = (int) o;

CONVERT

- Convierte un tipo de datos base en otro tipo
- Excepciones
  - InvalidCastException
    - No se admite una conversión determinada
    - Las conversiones de Char a Boolean, Single, Double, Decimal, o DateTime.
    - Las conversiones de Boolean, Single, Double, Decimal, o DateTime a Char.
    - Las conversiones de DateTime a cualquier otro tipo excepto String.
    - Conversiones de cualquier otro tipo, excepto String, a DateTime.
  - FormatException
    - Esto se produce cuando el intento de convertir un valor de cadena a cualquier otro tipo base porque la cadena no está en el formato correcto
  - OverflowException
    - Ocurre cuando una conversión de restricción produce una pérdida de datos. Por ejemplo, al intentar convertir una Int32 instancia cuyo valor es 10000

CONVERT

- **Convierte un tipo de datos base en otro tipo**
- **Se pueden realizar los siguientes tipos de conversiones:**
  - **Implícitas:**No se requiere una sintaxis especial porque la conversión es segura y no se perderán datos.Los ejemplos incluyen conversiones de tipos integrales de menor a mayor.

```
long valorGrande;
int entero = 5;
valorGrande = entero;
```

  - **Explícitas:** Interviene el programador, porque puede haber pérdida de datos. Ejemplo paso de un número grande a uno pequeño, se castea pero en algunos casos habrá perdida de datos.

```
double valorGrande=5555.5555;
int entero ;
 entero = (int) valorGrande;
```

CONVERT Y PARSE

- Cuando utilizamos la segunda opción, el método Parse(), si el usuario no ha introducido ningún valor (null) recibiremos una excepción del tipo System.FormatException. Lo que indica que el formato del argumento no cumple las especificaciones del parámetro del método invocado.
- En cambio, cuando utilizamos la primera opción, los métodos de la clase System.Convert, si el valor pasádo al método es null, el método ConovertToInt32() devolverá un valor numérico, devolverá cero en todos los casos que el valor de entrada sea null.

```
int valorA = Convert.ToInt32(valorUsuario);

int valorB = Int32.Parse(valorUsuario);
```

EL TIPO OBJECT

UAIOnline ultra

- Todas los tipos de datos heredan de Object.
- Admite todas las clases de la jerarquía de clases de .NET Framework y proporciona servicios de bajo nivel a las clases derivadas.
- Se trata de la clase base fundamental de todas las clases de .NET Framework; es la raíz de la jerarquía de tipos.
  - Equals
  - Finalize()
  - GetGashCode()
  - GetType()
  - ToString()

EL TIPO STRING

UAIOnline ultra

- Pertenece a System.String
- Almacena cadenas
- Lo nuevo de el tipo String
  - Tiene muchos constructores
    - Permiten una Secuencia de N caracteres similares
    - string cadena = “Caracteres”
  - Otras propiedades y métodos
    - Length: Numero de caracteres
    - Chars: devuelve un carácter situado en un índice.
- Formateando las salidas
  - string.Format("El valor de {0}", nombre variable);

EL TIPO DATE

UAIOnline ultra

- Espacio de nombre Using System.DateTime
- Operaciones con Fechas
  - Fecha.AddDays(-1)
  - Fecha.AddYears(5)
- Formatear Fechas
  - String.Format("{0:dd-MM-yyyy}",fecha)
  - Fecha.ToString("dd-MM-yyyy")
  - https://msdn.microsoft.com/en-us/library/8kb3ddd4(v=vs.110).aspx

GENERACIÓN DE NÚMEROS ALEATORIOS

UAIOnline ultra

- Clase Random
- Representa un pseudo generador de números random
- Produce una secuencia de números que cumplen con ciertos requisitos estadísticos para la aleatoriedad.

```
Random r = new Random();
r.Next(valor inicial, valor final);
```

ARRAYS

UAIOnline ultra

- System.Array
- **Estaticos**
  - Int[] array = new int[2]
- **Dinámicos**
  - Array.Resize(ref array,4);
- **Matiz** (Array multidimensional)
  - Int[,] matriz = new int[100,200]

INTRODUCCIÓN A LOS FORMULARIOS

UAIOnline ultra

- Windows Forms ("WinForms" para abreviar) es una biblioteca de clase GUI incluida con .NET Framework.
- Permite el desarrollo de aplicaciones de escritorio y móviles de Windows orientadas a .NET Framework .
- WinForms es principalmente impulsado por eventos . Una aplicación consta de varios formularios (mostrados como ventanas en la pantalla), que contienen controles (etiquetas, botones, cuadros de texto, listas, etc.) con los que el usuario interactúa directamente.

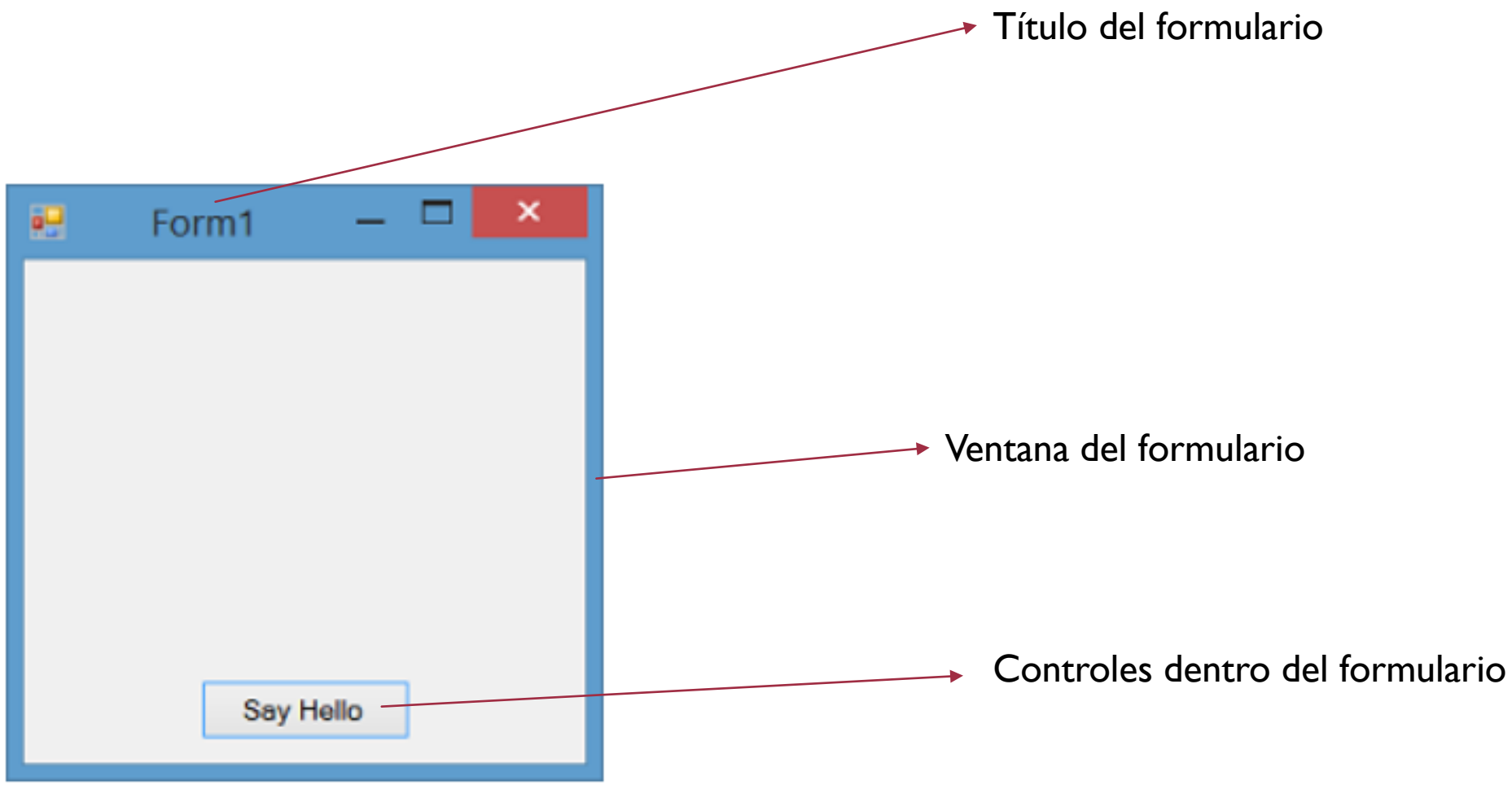
INTRODUCCIÓN A LOS FORMULARIOS

UAIOnline ultra

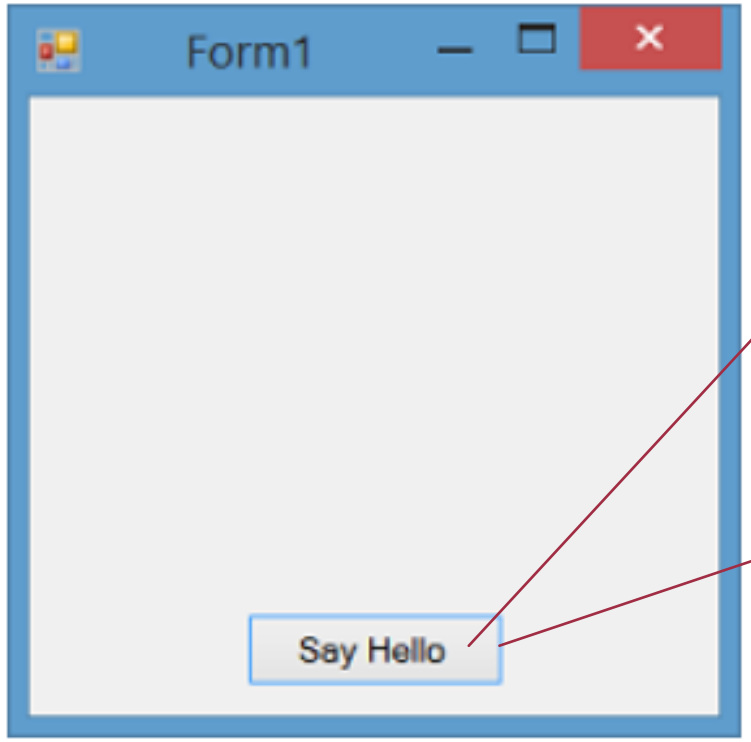
- Una aplicación consta de varios formularios (mostrados como ventanas en la pantalla), que contienen controles (etiquetas, botones, cuadros de texto, listas, etc.) con los que el usuario interactúa directamente.
- En respuesta a la interacción del usuario, estos controles generan eventos que el programa puede manejar para realizar tareas.
- Como en Windows, todo en WinForms es un control de usuario. La clase de **Control** proporciona una funcionalidad básica, que incluye propiedades para configurar texto, ubicación, tamaño y color, así como un conjunto común de eventos que se pueden manejar. Esto es común a todos los controles existentes.

INTRODUCCIÓN A LOS FORMULARIOS

UAIOnline ultra



INTRODUCCIÓN A LOS FORMULARIOS



Todos los controles (incluyendo las ventanas) tienen una serie de eventos que se pueden programar para lograr resultados esperados. Esto se hace a partir de funciones y procedimientos vinculadas al diseñador de formularios

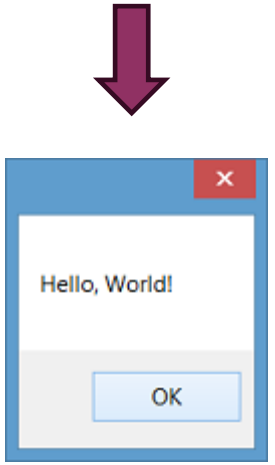
```
// When the button is clicked, display a message.
private void btnHello_Click(object sender, EventArgs e)
{
    MessageBox.Show("Hello, World!");
}
```

INTRODUCCIÓN A LOS FORMULARIOS

UAIOnline ultra

Clic

```
// When the button is clicked, display a message.
private void btnHello_Click(object sender, EventArgs e)
{
    MessageBox.Show("Hello, World!");
}
```



FORMULARIOS MDI

UAIOnline ultra

MENUES

UAIOnline ultra