

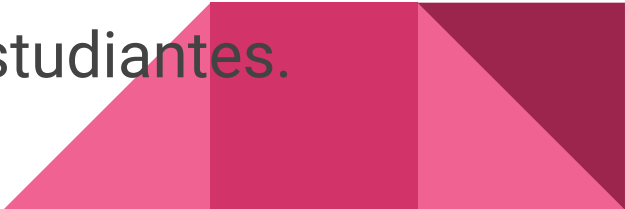
Taller de Sistemas de Información .NET

Práctico 1


Tecnólogo en informática - San José - 2023
Ing. Cristian Bauza

Introducción a Visual Studio

¿Qué es Visual Studio?

- Es un entorno de desarrollo integrado (IDE) desarrollado por Microsoft y que permite el desarrollo en múltiples Lenguajes.
 - Existen versiones pagas, pero tambien existe una version Community que puede ser usada por estudiantes.
- 

Introducción a Visual Studio

- Es posible desarrollar distintos tipos de proyectos, desde programas de consola hasta aplicaciones web, servicios web, aplicaciones móviles, entre otras.
 - En la estructura de un proyecto, estos siempre estarán contenidos en una “Solución”. Una “Solución” contiene uno o más proyectos, luego podemos referenciar los proyectos entre ellos para reutilizar funcionalidades.
- 

Introducción a Visual Studio

- Típicamente, en una aplicación empresarial, vamos a tener una solución que contiene varios proyectos, y mínimamente uno proyecto por cada capa de la arquitectura.



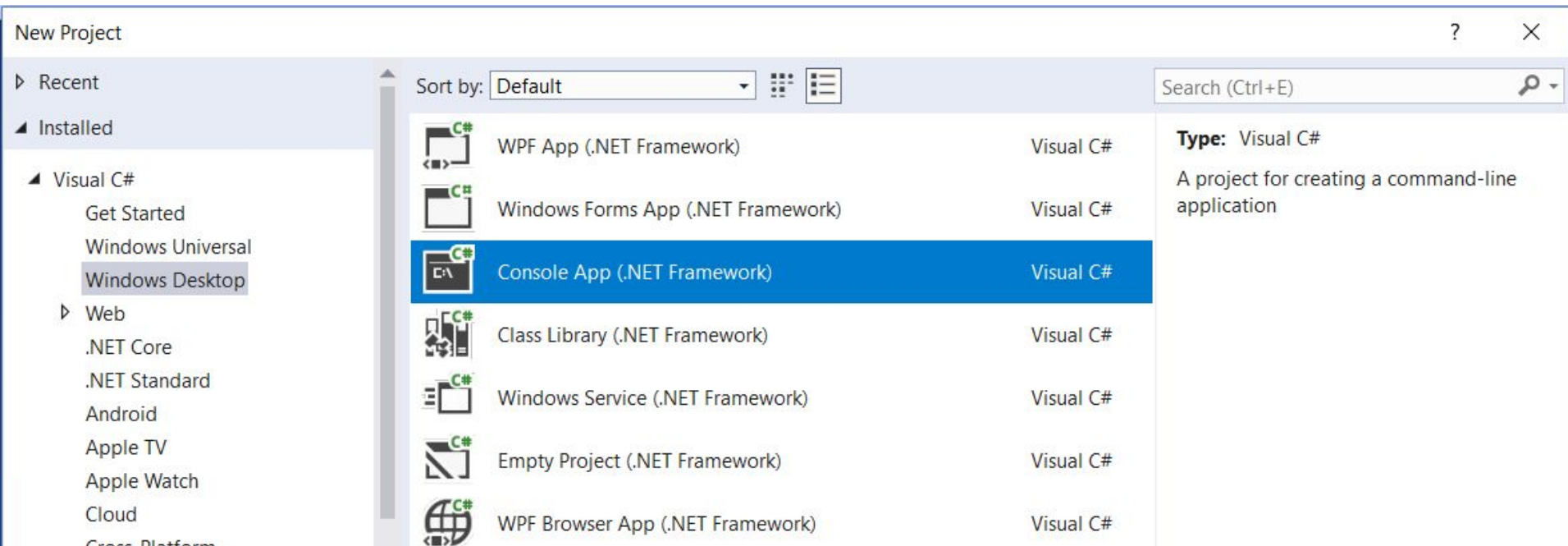
Introducción a Visual Studio

Vamos a crear una primera solución con distintas capas a modo de introducción.

- 1) Crear un nuevo proyecto, “Archivo -> Nuevo Proyecto”
- 2) Luego seleccionar “Aplicación de Consola” dentro de Visual C# → Aplicación de Escritorio de Windows



Introducción a Visual Studio



Introducción a Visual Studio

1) Agregar dentro de método Main:

```
Console.WriteLine("Mi Primera App");
```

```
Console.ReadLine();
```

Y luego ejecutar con:

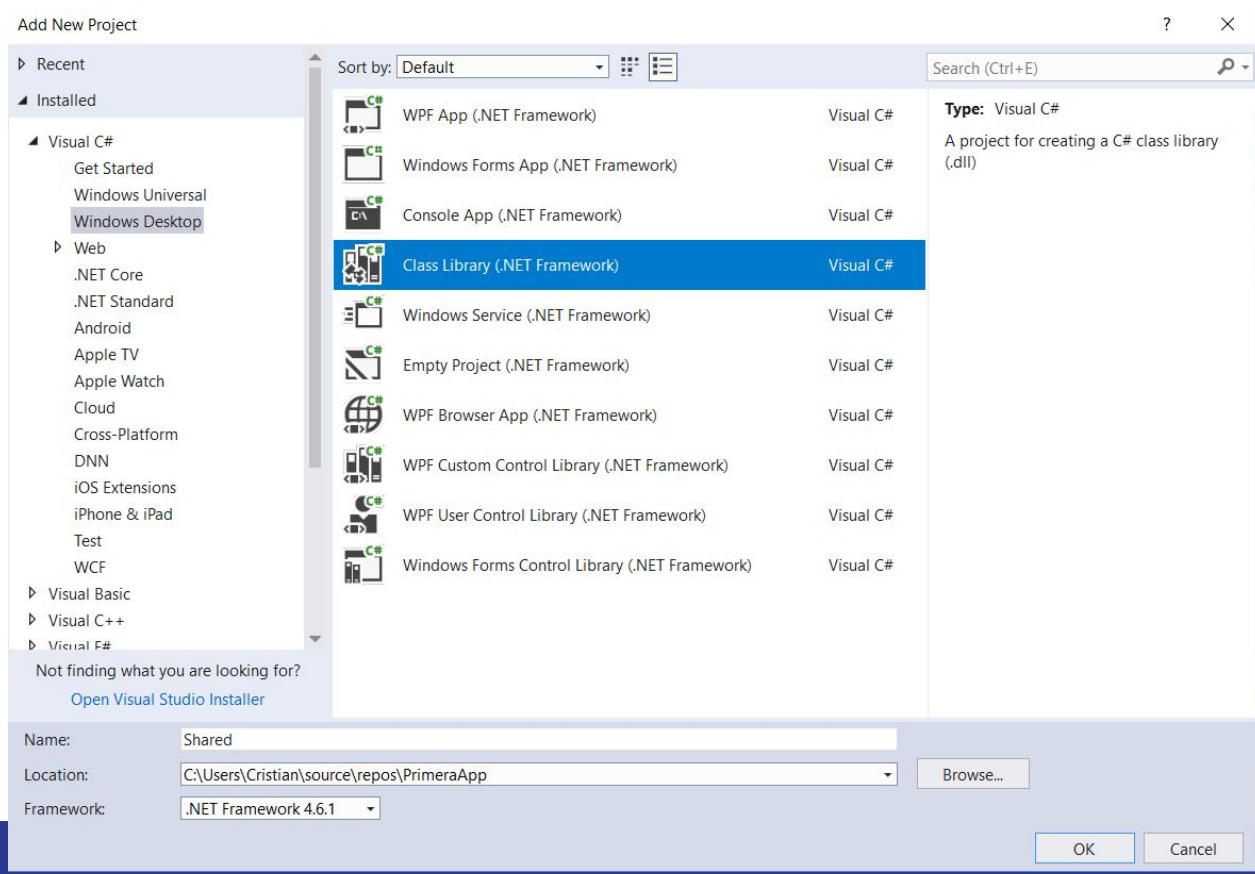


Introducción a Visual Studio

A continuación, hacer clic derecho sobre la solución y vamos a agregar un nuevo proyecto de nombre “Shared” y del tipo “Librería de Clases”. Este proyecto tendrá las entidades de negocio, y todo lo que sea transversal a las distintas capas.




Introducción a Visual Studio



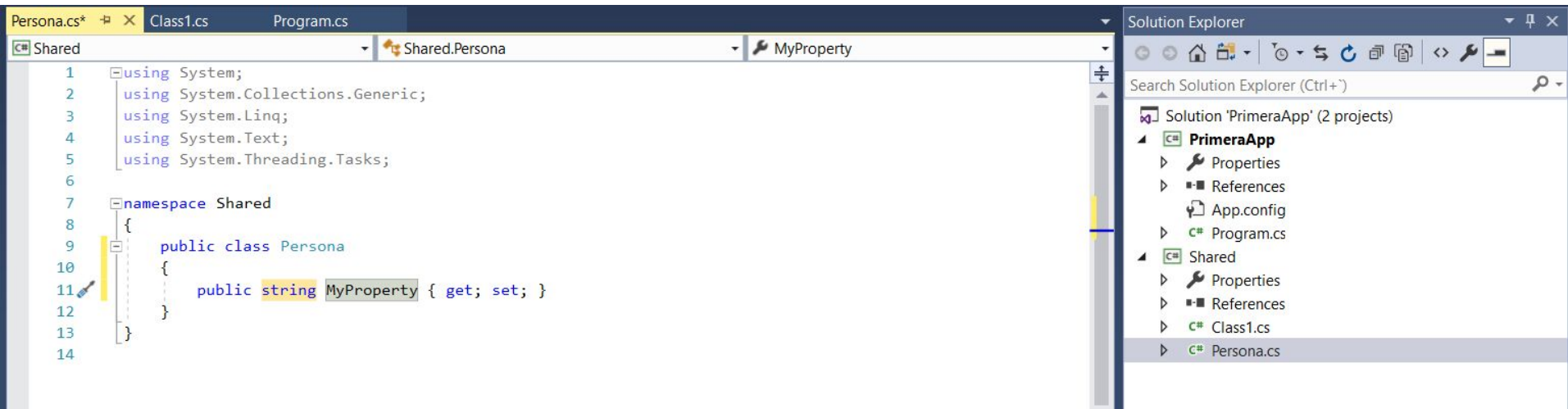
Introducción a Visual Studio

Luego dentro de esa librería de clases vamos a crear una clase Persona, esta clase tendrá un atributo nombre.

Para declarar una propiedad, podemos escribir dentro de la definición de la clase la palabra “prop” y luego apretando la tecla tabulador se definirá la propiedad y siguiendo con “tabulador” se posicionará en la selección para asignar tipo y nombre.



Introducción a Visual Studio



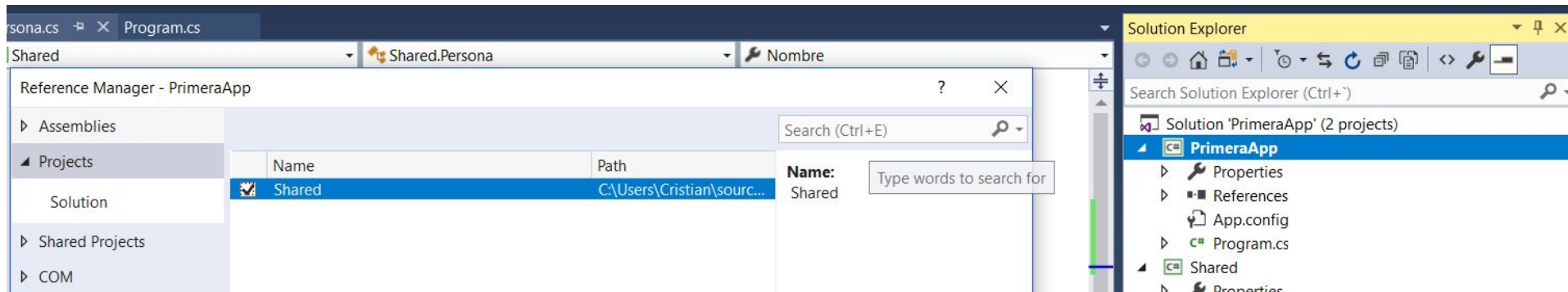
Introducción a Visual Studio

- Como se puede observar, por defecto se crean las propiedades como “autopropiedades”, si queremos realizar alguna validación, veremos más adelante como.

```
7 namespace Shared
8 {
9     public class Persona
10    {
11        public string Nombre { get; set; } = "-- Sin Nombre --";
12    }
13 }
14
```

Introducción a Visual Studio


Ahora vamos a agregar la referencia al proyecto “Shared” en el programa de consola, para ello hacemos click derecho, agregar, referencia, y luego elegimos referencia a proyecto:



Introducción a Visual Studio

Con esto logramos que la componente Shared sea visible desde el programa, entonces desde cualquier clase del programa de consola podremos hacer referencia a una clase de la componente Shared, y básicamente existen dos formas, una con referencia directa en cada línea, por ejemplo:

Shared.Persona o importante la componente en los imports de la clase: *using Shared;*



Introducción a C#

Como siguiente paso vamos a asignar la lectura de una línea de la entrada de la consola a la propiedad nombre de un objeto de la clase persona y luego vamos a hacer el print del resultado de esa línea.

```
10  class Program
11  {
12      static void Main(string[] args)
13      {
14          Console.WriteLine("Mi Primera App");
15          Persona per = new Persona();
16          per.Nombre = Console.ReadLine();
17          Console.WriteLine("Nombre: " + per.Nombre);
18          Console.ReadLine();
19      }
20  }
```

Introducción a C#

Propiedades con validación. Para validar una propiedad, se declara un atributo privado que almacena el valor de la propiedad y luego se realizan los controles correspondientes en los getters y los setters de la propiedad.



Introducción a C#

```
7 namespace Shared
8 {
9     public class Persona
10    {
11        public string Nombre { get; set; } = "-- Sin Nombre --";
12
13        private string documento = "";
14        public string Documento
15        {
16            get { return documento; }
17            set
18            {
19                if (value.Length < 7)
20                    throw new Exception("Formato de documento incorrecto.");
21                else
22                    documento = value.ToUpper();
23            }
24        }
25    }
26 }
27
```


Introducción a C#

```
14 Console.WriteLine("Mi Primera App");
15 do
16 {
17     try
18     {
19         Console.WriteLine("Nueva Persona");
20         Persona per = new Persona();
21         Console.Write("Nombre: ");
22         per.Nombre = Console.ReadLine();
23         Console.Write("Documento: ");
24         per.Documento = Console.ReadLine();
25
26         Console.WriteLine("Persona");
27         Console.WriteLine("    Documento: " + per.Documento);
28         Console.WriteLine("    Nombre: " + per.Nombre);
29     }
30     catch (Exception ex)
31     {
32         Console.WriteLine("ERROR: " + ex.Message);
33     }
34 }
35 while (!Console.ReadLine().Equals("EXIT"));
```

Introducción a C#

Uno de los aspectos más importantes de C# es el manejo de funciones Lambda. Podemos trabajar sobre colecciones con estas funciones y generar expresiones mucho más concretas.

Para considerar un ejemplo, vamos a declarar un array de enteros y vamos a aplicarle distintas funciones y obtener los resultados.



Introducción a C#

```
38 // Parte 2
39 int[] aux = { 2, 3, 6, 1, 0, 8, 9, 4, 5, 56, 10, 88, 23, 44, 11 };
40 aux.ToList().ForEach(x => Console.Write(x+"|"));
41 Console.WriteLine();
42
43 int[] aux2 = aux.ToList().OrderBy(x => x).ToArray();
44 aux2.ToList().ForEach(x => Console.Write(x + "|"));
45 Console.WriteLine();
46
47 int[] aux3 = aux.ToList().Where(x => x < 10).OrderBy(x => x).ToArray();
48 aux3.ToList().ForEach(x => Console.Write(x + "|"));
49 Console.WriteLine();
50
```

Introducción a C#

Ejercicio:

Implementar un programa de consola que registre datos de personas y al finalizar el registro con la palabra exit imprima todas las personas ordenadas por edad. La clase persona debe tener las propiedades Nombre, Apellido, Documento del tipo string, y FechaNacimiento de tipo DateTime.

