

10 conceptos que has aprendido en el curso

1- A crear objetos nuevos:

como en la imagen aparece así lo creamos `coche Ferrari = new coche`.

Y lo llamamos para ponerle nombre al coche nuevo usando `coche bmw = new coche()`;

Que también quieres llamar al nombre del coche pues le dices `bmw.(categoria)=`

(numero o palabras entre comillas, claro todo depende el constructor q tengas si es int o string);

```
5
6
7   coche Ferrari = new coche();
8
9   coche SeatPanda = new coche();
10  coche Renault = new coche();
11  coche bmw = new coche();
12  coche FordFiesta = new coche();
13  FordFiesta.NumeroPuertas = 5;
14  Ferrari.NumeroPuertas = 2;
15  SeatPanda.NumeroPuertas = 4;
16  bmw.NumeroPuertas = 4;
17  Renault.NumeroPuertas = 5;
18
19  FordFiesta.color = "rojo";
20  Ferrari.color = "blanco";
21  SeatPanda.color = "amarillo";
22  bmw.color = "azul";
23  Renault.color = "naranja";
24
25  FordFiesta.Matricula = "12";
26  Ferrari.Matricula = "23";
27  SeatPanda.Matricula = "34";
28  bmw.Matricula = "45";
29  Renault.Matricula = "56";
30
31
32
33
```

2- A crear constructores :

Como en la imagen muestra , creamos el public string Nombre {get; set;}

Y con ello ya podemos pedirle en el program el nombre, como en la segunda imagen.

```
3 referencias | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
public string Nombre { get; set; }

2 referencias | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
public string Apellidos { get; set; }

2 referencias | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
public string ColorPelo { get; set; }

2 referencias | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
public double Estatura { get; set; }

1 referencia | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
public string Dni { get; set; }

1 referencia | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
public int Telefono { get; set; }

private string NumeroSeguridadSocial;

1 referencia | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
public void EstablecerNss(string Nss)
{
    NumeroSeguridadSocial = Nss;
}

1 referencia | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
public string RecuperarNss()
{
    return NumeroSeguridadSocial;
}

Persona juan = new Persona();
juan.Nombre = "Juan";
juan.Apellidos = "Lopez";
juan.ColorPelo = "Rubio intenso numero 7";
juan.Estatura = 1.80;
juan.Dni = "12345678G";
juan.Telefono = 916019999;

Persona Mario = new Persona();
Mario.Nombre = "Mario";
Mario.Apellidos = "Martinez";
Mario.ColorPelo = "Marron";
Mario.Estatura = 1.65;

Persona Maria = new Persona();
Maria.Nombre = "Maria d ela O";

Animal Tigre = new Animal();
Tigre.Nombre = "Huevon";
Tigre.Identificacion = 12345;

Animal Perro = new Animal();
Perro.Nombre = "Lucas";
Perro.Identificacion = 234567;

0 referencias | - cambios | -autores, - cambios
static void Main(string[] args)
{
    // ...
}
```

3- Que es un método?

Un método puede ser todo aquel que representa la vista al **publico o al privado**

(public, private, protected, internal).

```
using System;

namespace MiCalculadoraProyecto
{
    0 referencias | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
    internal class Program
    {
        /*
        static void Main(string[] args)
        {
            Calculadora calculadora = new Calculadora();
        }
        */
    }
}
```

4- Que son parámetros?

Son los valores con los que se definen los métodos, en la imagen muestro como **int sumando1, int sumando2 o int minuendo, int sustraendo**.

```

using System;

namespace MiCalculadoraProyecto
{
    2 referencias | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
    public class Calculadora
    {
        1 referencia | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
        public int Suma(int sumando1, int sumando2)
        {
            return sumando1 + sumando2;
        }

        1 referencia | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
        public int Resta(int minuendo, int sustraendo)
        {
            return minuendo - sustraendo;
        }

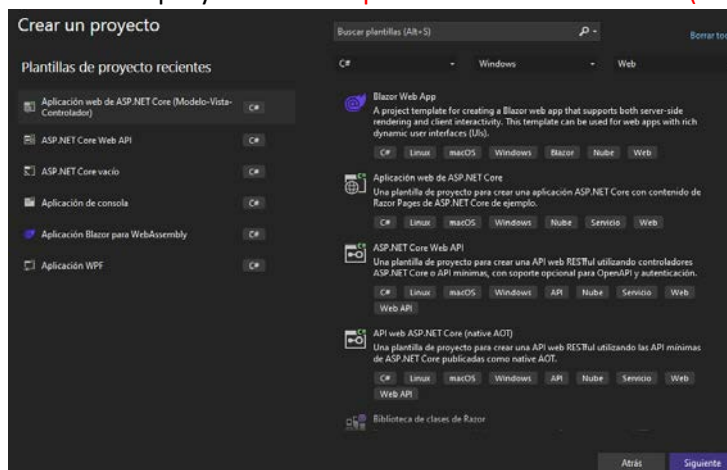
        1 referencia | pedrojoeverdugo, Hace 20 días | 1 autor, 1 cambio
        public int Multiplicar(int factor1, int factor2)
        {
            return factor1 * factor2;
        }
    }
}

```

5- A generar scaffolding automaticos para asp en paginas web

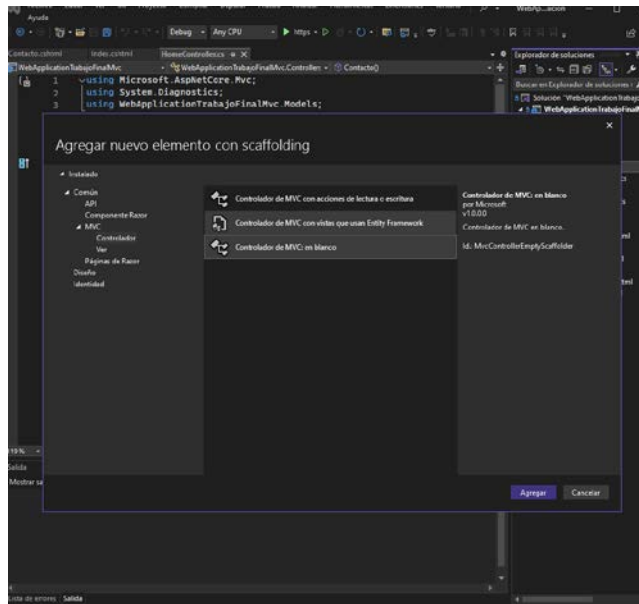
Esto nose como ponerlo....

Creamos un proyecto con la aplicacion web ASP.Net core(modelo-vista-controlador).



6- Crear controladores en modo (modelo-vista-controlador)

Pinchamos en controlador, en agregar , controlador.



7- A agregar al controlador nuevas vistas

Como el index y el contacto q aparece en la imagen

```

4
5 namespace WebApplicationTrabajoFinalMvc.Controllers
6 {
7     3 referencias | pedroJoseverdugo, Hace 3 días | 1 autor, 1 cambio
8     public class HomeController : Controller
9     {
10         private readonly ILogger<HomeController> _logger;
11
12         0 referencias | pedroJoseverdugo, Hace 3 días | 1 autor, 1 cambio
13         public HomeController(ILogger<HomeController> logger)
14         {
15             _logger = logger;
16
17         0 referencias | pedroJoseverdugo, Hace 3 días | 1 autor, 1 cambio
18         public IActionResult Index()
19         {
20             return View();
21         }
22
23         0 referencias | pedroJoseverdugo, Hace 3 días | 1 autor, 1 cambio
24         public IActionResult Contacto()
25         {
26             return View();
27         }
28     }

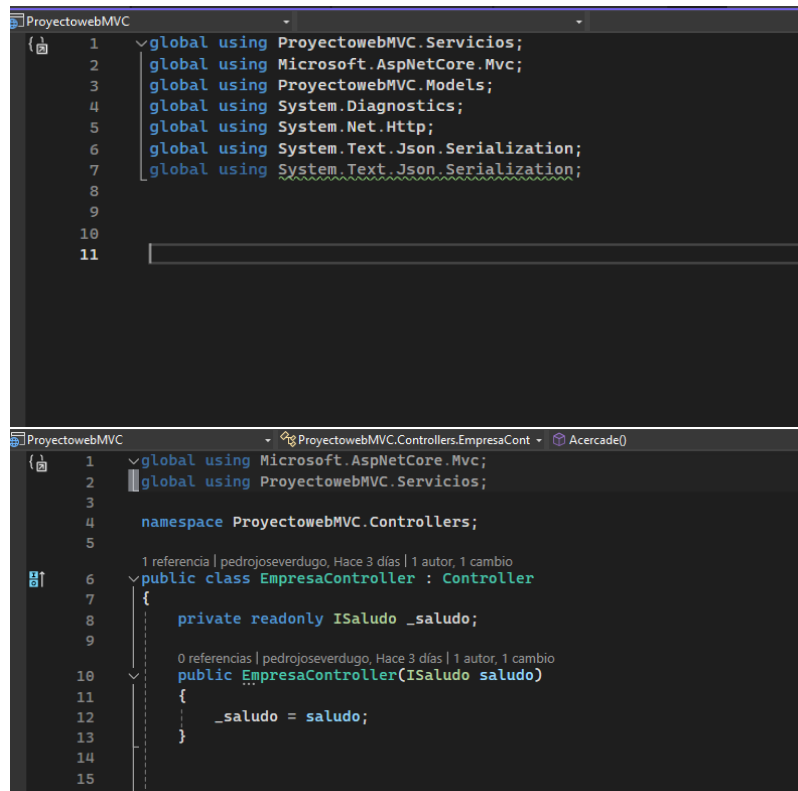
```

8- A crear una vista desde el controlador:

dentro del controlador ,encima de la palabra index botón derecho agregar vista y ahí ya nos crea la carpeta de la vista y la vista como tal

9- A usar una herramienta super buena llamada Global using

Creamos una **pestaña nueva** parecida a program en la **misma altura que program** y le ponemos el **global using** de todos los using que tenemos en todos los controladores. A y también tenemos q **ponerle global using y la extensión del using** que tenemos, como en la imagen indica



The image consists of two screenshots of the Visual Studio code editor. The top screenshot shows a file named 'ProyectoWebMVC' with a list of global using statements at the top of the file. The bottom screenshot shows a file named 'ProyectoWebMVC.Controllers.EmpresaCont' with a list of global using statements at the top of the file, followed by a namespace declaration and a class definition.

```
1 global using ProyectoWebMVC.Servicios;  
2 global using Microsoft.AspNetCore.Mvc;  
3 global using ProyectoWebMVC.Models;  
4 global using System.Diagnostics;  
5 global using System.Net.Http;  
6 global using System.Text.Json.Serialization;  
7 global using System.Text.Json.Serialization;  
8  
9  
10  
11
```

```
1 global using Microsoft.AspNetCore.Mvc;  
2 global using ProyectoWebMVC.Servicios;  
3  
4 namespace ProyectoWebMVC.Controllers;  
5  
6 public class EmpresaController : Controller  
7 {  
8     private readonly ISaludo _saludo;  
9  
10    public EmpresaController(ISaludo saludo)  
11    {  
12        _saludo = saludo;  
13    }  
14  
15
```

10- Usar interface

Ponemos **public interface ISaludo** y lo q quieras poner de constructor y de método (como en la imagen que os pongo a continuación...)

```
ProjectowebMVC
  ProjectowebMVC.Servicios.ISaludo
    Saludo()
1  namespace ProjectowebMVC.Servicios;
2
3  5 referencias | pedrojoeverdugo, Hace 3 días | 1 autor, 1 cambio
4  public interface ISaludo
5  {
6
7  2 referencias | pedrojoeverdugo, Hace 3 días | 1 autor, 1 cambio
8  string Mensaje { get; set; }
9  4 referencias | pedrojoeverdugo, Hace 3 días | 1 autor, 1 cambio
10 public string Saludo();
11
12 }
```

Y la llamamos en la principal de saludo con **public class SaludoAleman : ISaludo** y agregamos el **public string Saludo()** y el retorno deseado como a continuación...

```
ProjectowebMVC
  ProjectowebMVC.Servicios.SaludoEnAlema
  Mensaje
1  namespace ProjectowebMVC.Servicios;
2
3  1 referencia | pedrojoeverdugo, Hace 3 días | 1 autor, 1 cambio
4  public class SaludoEnAleman : ISaludo
5  {
6  1 referencia | pedrojoeverdugo, Hace 3 días | 1 autor, 1 cambio
7  public string Mensaje { get; set; }
8  3 referencias | pedrojoeverdugo, Hace 3 días | 1 autor, 1 cambio
9  public string Saludo()
10 {
11
12     return "Hallo Welt";
13 }
14 }
```