ASP .NET MVC DATAANNOTATIONS

¿QUÉ ES DATAANNOTATIONS?

- **DataAnnotations se encuentra en el espacio de nombres System.ComponentModel**
- **Se utiliza para configurar las propiedades de nuestras clases. Normalmente de nuestras clases viewModel o en nuestras clases de persistencia.**
- **Una de las aplicaciones más comunes es para validar.**

VALIDACIONES CON DATAANNOTATIONS

- Las validaciones son importantes para asegurarnos que los datos introducidos por un usuario son correctos.
- .NET nos ofrece una manera muy sencilla para validar.
- Para hacerlo, hace uso del ModelState.

ModelState

- El ModelState es un objeto, gestionado por el framework de ASP.NET MVC que indica si el modelo es correcto o no.
- El modelo es el objeto que recibe un controlador.
- **Correcto** significa que los datos de la petición eran válidos y que el Model Binder ha podido crear un objeto y rellenarlo.
- **Incorrecto** significa que había algún dato de la petición inválido, ya sea por algún error (una cadena alfanumérica se ha intentado asignar a un int) o bien por alguna validación fallida (una cadena declarada como email no tenía el formato correcto).

¿Cómo usamos el ModelState?

```
[HttpPost]
    public ActionResult Index(clsPersona oPersona)
    {
        IActionResult action;
       if (!ModelState.IsValid)
       {
        //Mandamos el objeto a la misma vista (un formulario)
        //para que se muestren los mensajes de error
           action=View(oPersona);
        }
       else
       {
          // Todo Correcto
           //Podemos mandar a una página de éxito
                 action=View("vistaExito");
            //O podemos mandar a la misma vista pero con un ViewBag
          …
        }
          return action;
    }
```

¿Cómo configuramos la clase?

```
public class clsPersona
    {
      public int idPersona { get; set; }
        [Required(ErrorMessage = “Campo obligatorio”)]
[Display(Name=“Nombre")]
public string nombre { get; set; }
[MaxLength(50),Required]
public string apellidos { get; set; }
        [DataType(DataType.Date)]
    [DisplayFormat(DataFormatString="{0:dd-MM-yyyy}", ApplyFormatInEditMode=true)]
      public DateTime fechaNac { get; set; }
      public string direccion { get; set; }
      public string telefono { get; set; }

    }
```

Algunos atributos

- **DataType:** especifica el tipo de la propiedad.
- **DisplayName:** especifica el nombre que será visible con @HtmlDisplayNameFor(model=>model.nombre).
- **DisplayFormat:** especifica el formato que se mostrará.
- **Required:** especifica que la propiedad es obligatoria.
- **ReqularExpression:** permite escribir una expression regular para validar la propiedad.
- **Range:** permite especificar un rango para la propiedad.
- **StringLength:** permite definer un máximo y mínimo de caracteres.
- **MaxLength:** permite definer un máximo de caracteres.
- **Bind:** especifica las propiedades que queremos enlazar de mi clase:

```
[Bind(Include="idPersona, nombre")]
public class clsPersona
        {
```

¿Cómo se muestran los errores?

- Podemos hacerlo de dos formas:
  - @Html.ValidationMessageFor(model=> model.nombre)
  - @Html.ValidationSummary()

ValidationMessageFor

- Se usa para mostrar en la vista el error de cada una de las propiedades.
- El error mostrado será el que hemos especificado en errorMessage.
- Si no hemos especificado ninguno, aparece un mensaje por defecto.

```
<p>
    @Html.TextBoxFor(model => model.Edad)
    @Html.ValidationMessageFor(model => model.Edad)
</p>
```

ValidationSummary

- Muestra una lista con todos los errores cometidos.
- Genera una <ul> con un <li> por cada error de cada propiedad.

@Html.ValidationSummary()