Desarrollo de Web Api’s con .NET

ASP.NET Core MVC

¿Qué vamos a ver hoy?

- Patrón MVC
- Repartición de responsabilidades
- Enrutamientos
- Razor

Arquitectura **M**odel-**V**iew-**C**ontroller

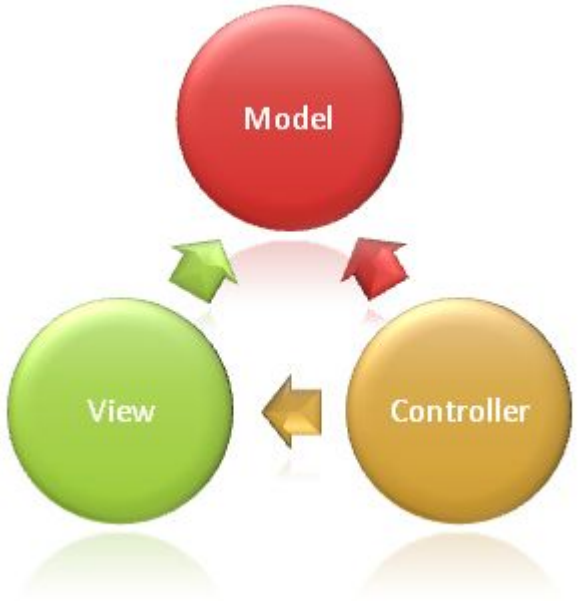
Separa una aplicación en tres grupos de componentes principales: modelos, vistas y controladores. De esta manera, es más difícil actualizar, probar y depurar código que tenga dependencias repartidas.

Por ejemplo, la lógica de la interfaz de usuario tiende a cambiar con mayor frecuencia que la lógica de negocios. Si el código de presentación y la lógica de negocios se combinan en un solo objeto, un objeto que contenga lógica de negocios deberá modificarse cada vez que cambie la interfaz de usuario.

Arquitectura **M**odel-**V**iew-**C**ontroller

La vista y el controlador dependen del modelo.

El modelo no depende de la vista ni del controlador.



Responsabilidades del **modelo**

El modelo representa el estado de la aplicación y cualquier lógica de negocios u operaciones que esta deba realizar.

La lógica de negocios debe encapsularse en el modelo, junto con cualquier lógica de implementación para conservar el estado de la aplicación.

Las vistas fuertemente tipadas normalmente usan tipos ViewModel diseñados para que contengan los datos para mostrar en esa vista. El controlador crea y rellena estas instancias de ViewModel desde el modelo.

Responsabilidades de la **vista**

Las vistas se encargan de presentar el contenido a través de la interfaz de usuario.

Debería haber la mínima lógica entre las vistas, y cualquier lógica en ellas debe estar relacionada con la presentación de contenido.

Si se necesita realizar una gran cantidad de lógica en los archivos de vistas para mostrar datos de un modelo complejo, deberíamos delegar cierta responsabilidad en un ViewModel o una plantilla de vista para simplificar la vista.

Responsabilidades del **controlador**

Los controladores son los componentes que controlan la interacción del usuario, trabajan con el modelo y seleccionan una vista para representarla.

Es quien controla y responde a la interacción y los datos que introducen los usuarios.

Es el punto de entrada inicial que se encarga de seleccionar con qué tipos de modelo trabajar y qué vistas representar.

Algunas sugerencias...

Los controladores no deberían complicarse con demasiadas responsabilidades.
Para evitar que la lógica del controlador se vuelva demasiado compleja, sacamos la lógica de negocios fuera el controlador y la llevamos al modelo de dominio.

Si vemos que el controlador realiza con frecuencia los mismos tipos de acciones, podemos poner estas acciones comunes en filtros.

Enlace de modelos

MVC convierte los datos de solicitud de cliente (valores de formulario, datos de ruta, parámetros de cadena de consulta, encabezados HTTP) en objetos que el controlador puede controlar.

Como resultado, la lógica del controlador no tiene que realizar el trabajo de pensar en los datos de la solicitud entrante; simplemente tiene los datos como parámetros para los métodos de acción.

Enlace de modelos

C#

```
using System.ComponentModel.DataAnnotations;
public class LoginViewModel
{
    [Required]
    [EmailAddress]
    public string Email { get; set; }

    [Required]
    [DataType(DataType.Password)]
    public string Password { get; set; }

    [Display(Name = "Remember me?")]
    public bool RememberMe { get; set; }
}
```

¡¡A trabajar!!