![Diagrama

Descripción generada automáticamente]()

**Las siglas MVC:**

Corresponden al término Modelo-Vista-Controlador, forma en que se denomina a uno de los patrones de arquitectura más populares. En *MVC* se pretende separar completamente la información que gestiona nuestra aplicación (el Modelo) de la forma en que se presenta (la Vista), desacoplando ambas capas mediante una capa intermedia (el Controlador) que contendrá toda la lógica para trasladar los datos a la presentación visual que deseamos realizar.

![Diagrama

Descripción generada automáticamente]()

**El modelo**

Forma parte de la aplicación que es responsable de la aplicación básica o la lógica comercial. Los objetos de modelo normalmente obtienen acceso a los datos desde un almacén persistente, como SQL Server, y realizan la lógica comercial en esos datos. Los modelos son específicos de la aplicación.

Por ejemplo, el marco de ASP.NET MVC no impone ninguna restricción sobre los tipos de objetos de modelo que se pueden generar. Puede utilizarse objetos DataReader o DataSet de ADO.NET o puede utilizarse un conjunto personalizado de objetos de dominio. También puede utilizar una combinación de tipos de objeto para trabajar con datos.

Crear un modelo simple para una persona ASP.NET

<https://youtu.be/MnTrXgPxgtg>

**Controladores y métodos de acción en aplicaciones ASP.NET - MVC**

Los controladores procesan solicitudes entrantes, controlan los datos proporcionados por el usuario y las interacciones y ejecutan la lógica de la aplicación adecuada. Una clase de controlador llama normalmente a un componente de vista independiente para generar el marcado HTML para la solicitud.

Todas las clases de controlador deben llevar el sufijo "Controller" en su nombre. En el ejemplo siguiente se muestra la clase de controlador de ejemplo, que se denomina HomeController. Esta clase de controlador contiene métodos de acción que representan las páginas de vista.

**¿Qué es el resultado de la acción?**

El resultado de la acción es básicamente el tipo de retorno o el resultado de su método de acción. ActionResult es la clase base para todos los resultados de acciones. El espacio de nombres para esto es *System.Web.MVC.*

Si necesita devolver un resultado de acción múltiple, puede usar ActionResult como tipo de retorno. La siguiente tabla muestra los tipos de resultados de acción incorporados y los métodos auxiliares de acción que los devuelven.

Crear un controlador para el modelo Personal.

<https://youtu.be/A3U-zk_mSXQ>

**Crear una vista de ASP.NET MVC llamando a varias acciones**

La manera típica de crear una vista en *ASP.NET MVC* es llamar a un método de acción que utiliza el modelo para preparar los datos de la vista. A continuación, el método de acción llama al método View del controlador para crear la vista. Sin embargo, puede ser conveniente llamar a métodos de acción diferentes para crear las distintas partes de una vista. Por ejemplo, puede haber una página que muestre las últimas noticias, el tiempo y la información deportiva. En ese caso, podría ser útil tener un método de acción que controle las noticias, otro para el tiempo y un tercero para los deportes. De este modo, es posible presentar vistas distintas en condiciones diferentes o desglosar vistas complejas en elementos más pequeños y fáciles de controlar.

<https://youtu.be/JCrIE7OC0EQ>

**Diferencias entre MVC y Webforms**

Ambas tecnologías tienen sus pros y sus contras. Depende totalmente de los requisitos del proyecto si *ASP.NET webforms* o *MVC* se adapta a las necesidades.

MVC que significa Modelo-Vista-Controlador es un patrón de diseño arquitectónico que separa la aplicación en tres unidades lógicas (Modelo (Datos / Entidades de Negocio), Vista (UI), Controlador (Manejador de solicitudes))

![Diagrama

Descripción generada automáticamente]()

**Principales diferencias entre Webforms y MV:**

ASP.NET Webforms (Desarrollo basado en eventos; Uso de los controles del servidor para el diseño de la página, admite el estado de la vista para la gestión del estado; URLs basadas en archivos, se requiere la existencia física del archivo, por ejemplo, para la URL, habrá un archivo físico index.aspx en el servidor; los archivos de diseño aspx están vinculados a los archivos de código aspx.cs; Necesidad de reescribir las URLs para que sean amigables con el SEO)

Mientras que ASP.NET MVC (Desarrollo basado en patrones de diseño MVC y no en eventos; No hay controles de servidor, sino que se utilizan controles y ayudantes HTML; No hay concepto de estado de vista; URLs basadas en rutas, por lo que no se requiere la existencia física del archivo; No hay acoplamiento estrecho, un controlador puede servir para diferentes vistas; Las URLs amigables con el SEO pueden ser fácilmente formadas sin una compleja reescritura de URLs.)

Diagrama

Descripción generada automáticamenteOtro ejemplo gráfico arquitectura MVC de una aplicación web CRUD de entidades básicas para un restaurante como Producto, Marca, Unidad de Medida, Cliente, Rol, Usuario. Arquitectura MVC