

## Teoría

1) Verdadero o falso ASP.NET MVC. Justifique en los casos que sea falso:

a) Todas las clases que representan al Model, deben terminar con la palabra "Model" por convención.

rta: Falso, no es necesario que terminen en Model puede no terminar en model y el proyecto puede funcionar bien, es una convención. ✓

b) El modo de enrutamiento de la aplicación se define en el archivo appsettings.json.

rta: Falso el enrutamiento de la aplicación se define en el archivo [Program.cs](#) que está en la raíz del proyecto o en versiones antiguas [Startup.cs](#). ✓

c) Un middleware es configurado en el método "Configure" de la clase "Startup". Solo puede existir un middleware por aplicación Web.

rta: Verdadero ✗

*Corrección:*

FALSO. En el método [Configure](#) (en [Startup.cs](#)) o en [Program.cs](#) se configuran múltiples middlewares. No hay límite en la cantidad de middlewares, y cada uno puede hacer tareas como autenticar, registrar logs, manejar errores, etc.

d) El controlador ejecuta lógica pero el Modelo es el responsable de elegir la vista que será procesada y devuelta al cliente.

rta: Falso el modelo no elige qué vista mostrar eso lo hace el controlador que llama a los métodos del modelo y dice qué vista debe mostrar. ✓

2) Dado un controlador con una única acción para el sitio [www.misitio.com](#) acción:

```
public class ProductosController : Controller
{
    public ActionResult MostrarTodosProductos()
    {
        return View(ProductosServicio.ObtenerTodosProductos());
    }
}
```

Y considerando la configuración de tabla de ruteo default de MVC mencionada en clase, indique cuáles de las siguientes URLs son válidas/correctas (no arrojarían error y encuentran una acción correspondiente en el controlador). Justifique muy bien su respuesta:

a) [http://www.misitio.com/ProductosController/MostrarTodosProductos](#)

b) [http://www.misitio.com/ProductosController/MostrarTodosActionProductos](#)

c) [http://www.misitio.com/Productos/MostrarTodos](#)

d) [http://www.misitio.com/Productos/MostrarTodosProductosAction](#)

e) [http://www.misitio.com/Productos/MostrarTodosProductos/ObtenerTodosProductos](#)

f) Ninguna es válida

rta: C y E es valida ya que cumple con los requisitos de la URL ,en la URL solo se muestra el nombre del controlador en este caso Productos y no ProductosController y en los metodo solo mostrar el nombre del metodo y no el Action como Productos y se puede seguir con / nombre de otro metodo. ❌

Corrección:

Correcta: f) Ninguna es válida

a) <http://www.misitio.com/ProductosController/MostrarTodosProductos>

❌ Inválida.

- Usa [ProductosController](#), pero el framework espera solo [Productos](#) como nombre del controlador.
- Justificación correcta: el enrutamiento espera el nombre sin "Controller".

b) <http://www.misitio.com/ProductosController/MostrarTodosActionProductos>

❌ Inválida.

- Mismo error que (a): [ProductosController](#) no es válido.
- Además, [MostrarTodosActionProductos](#) no existe como acción.

c) <http://www.misitio.com/Productos/MostrarTodos>

❌ Inválida.

- El método se llama [MostrarTodosProductos](#), no [MostrarTodos](#).
- Aunque el controlador está bien ([Productos](#)), el nombre del método no coincide.

d) <http://www.misitio.com/Productos/MostrarTodosProductosAction>

❌ Inválida.

- El método correcto es [MostrarTodosProductos](#), sin "Action".
- No existe una acción con ese nombre.

e) <http://www.misitio.com/Productos/MostrarTodosProductos/ObtenerTodosProductos>

❌ Inválida.

- Solo existe **una acción** llamada [MostrarTodosProductos](#).
- [/ObtenerTodosProductos](#) no corresponde a ningún parámetro ni acción adicional.

3) Verdadero o falso ASP.NET - MVC. Justifique en los casos que sea falso:

a) Respecto a las variables de sesion en ASP.NET CORE, existe un mecanismo automatico para informar al codigo de la aplicacion (servidor) que el navegador se ha cerrado.

rta: Verdadero ❌

*Corrección:*

FALSO. El servidor no tiene forma directa de saber si el navegador se cerró. Las sesiones expiran después de un período de inactividad.

b) El tag helper <partial name="\_Vista" /> es la forma tradicional de renderizar una vista parcial.

rta: Verdadero ✅

c) En ASP.NET CORE el uso de variables de sesion esta activado por defecto; solo es necesario utilizar el objeto o clase:

"HttpContext.Session".

rta: Falso para utilizar la clase hay que importarla con la palabra reservada using y el nombre de su libreria. ✅ y ❌

*Corrección:*

La sesión **no está activada por defecto** en ASP.NET Core. Además de importar la clase, se debe **configurar explícitamente** en [Program.cs](#) o [Startup.cs](#).

Se necesita agregar y configurar la sesión en el [Program.cs](#) usando [services.AddSession\(\)](#) y [app.UseSession\(\)](#)

d) ViewBag es similar a ViewData, pero se diferencian en su "tiempo de vida", así ViewData permite mantener variables y su contenido despues de una redireccion, pero ViewBag no.

rta: Verdadero ❌

*Corrección:*

FALSO. Tanto **ViewBag** como **ViewData** solo viven durante una solicitud. Si se hace una redirección, se pierde el contenido.

4) Explique y ejemplifique el 4 principio SOLID (I)

Rta: NO SE

*Corrección:*

RTA: **Interface Segregation Principle (ISP)**

Los clientes no deberían verse obligados a depender de interfaces que no utilizan.

Es mejor tener muchas interfaces específicas en lugar de una sola interfaz general con muchos métodos que pueden no ser utilizados por todas las clases.

// MALO: una interfaz grande con métodos innecesarios

```
interface ITrabajador
```

```
{  
    void Trabajar();  
    void Comer(); // ¿Y si un robot no come?  
}
```

// MEJOR:

```
interface ITrabajador
```


```
{  
    void Trabajar();  
}
```

```
interface IComedor
```


```
{  
    void Comer();  
}
```

5) Indique V o F. Justifique si es falso


a) En una aplicación ASP.NET MVC el runtime (Framework o Core) debe estar instalado en el server y se recomienda que también esté instalado en el cliente para evitar problemas de compatibilidad de versiones.

rta: Falso solo debe estar instalado en el servidor 

b) Un Assembly (.dll / .exe) es la unidad mínima de ejecución cuyo código MSIL es creado por el CLR.

rta: Verdadero 

c) .Net Core tiene similitudes a .Net Framework y fue escrito sobre la base de .Net Framework.

rta: Verdadero 

*Corrección:*

FALSO. .NET Core fue desarrollado desde cero como una plataforma modular, multiplataforma y de código abierto, independiente de .NET Framework.

d) En .Net para castear un string a un int a un string se puede usar la forma de conversion:

```
int i = (int) mystring;
```

rt:Verdadero ❌

*Corrección:*

FALSO. Para convertir un `string` a `int` se debe usar

`int.Parse(mystring)` o `Convert.ToInt32(mystring)`. El cast `(int)` solo funciona con tipos compatibles.