Practico 3 –

BetaGamer: Aficionados a los Juegos.

# Introducción

En este práctico crearemos una aplicación para el sitio “BetaGamer” que permite que aficionados a los videojuegos comenten sus juegos favoritos.

* Para esto definiremos dos clases: Juego y Comentario.
* Vamos a agregar soporte para Entity Framework
* Vamos a crear una clase de contexto que herede de DbContext para manejar las clases y el acceso a datos
* Vamos a construir una aplicación básica de MVC que nos permita ver, agregar, editar y borrar Juegos.

# Crear la Aplicación MVC

En el Visual Studio:

🡪Agregar🡪Nuevo Proyecto 🡪 ASP.NET Web Application, plantilla MVC sin test unitarios ni despliegue en la nube y configurar la autenticación de usuarios como sin autenticación.

🡪Nombre: BetaGamer

Visual studio va a crear toda la estructura del proyecto y carpetas con clases controladoras, modelos y vistas pre-definidas (Si corriéramos el proyecto mostraría una página de bienvenida).

# Crear el Modelo

Vamos a crear las clases de dominio. En el explorador de Soluciones:

🡪Click derecho sobre la carpeta Models🡪 Agregar🡪Clase🡪Nombre: Juego

🡪Click derecho sobre la carpeta Models🡪 Agregar🡪Clase🡪Nombre: Comentario

public class Juego

{

public int Id { get; set; }

public string Titulo { get; set; }

public DateTime FechaPublicacion { get; set; }

public string Descripción { get; set; }

public ICollection<Comentario> Comentarios { get; set; }

}

public class Comentario

{

public int Id { get; set; }

public DateTime Fecha { get; set; }

public string Texto { get; set; }

public Juego Juego { get; set; }

}

# INCLUIR ENTITY FRAMEWORK EN EL PROYECTO

VERIFICAREMOS SI ENTITY FRAMEWORK VERSIÓN 5.0.0 YA ESTÁ INCLUIDO EN EL PROYECTO Y SI NO, LA INSTALAREMOS:

* Clic derecho sobre el proyecto 🡪 Administrador de paquetes Nuget 🡪 Pestaña Instalados
* Escriba “entity” en el buscador y luego de dar Enter espere a que aparezcan los resultados de la búsqueda
* Si no aparece Entity Framework, abra la pestaña Navegar (Browse) y busque Entity Framework. Selecciónelo en el panel de la derecha y en el panel de la derecha elija la versión 5.0.0 del combo desplegable y haga clic en Instalar.
* Si la versión que aparece como instalada es la 6.x.x entonces deberemos desinstalarla primero y luego instalar la 5.0.0, ya que la versión 6 de Entity Framework no es compatible con la versión 4 de MVC

PUEDE HACER LAS MISMAS COMPROBACIONES, DESINTALACIÓN E INSTALACIÓN MEDIANTE COMANDOS USANDO LA CONSOLA DE ADMINISTRACIÓN DE PAQUETES:

* Click en Herramientas 🡪 Administrador de paquetes Nuget 🡪 Consola de administración de paquetes
* Para conocer la versión instalada actualmente se puede ejecutar el comando:

Get-Package EntityFramework

* Para Desinstalar la versión 6:

UnInstall-Package EntityFramework -Version <version number>

* Para Instalar la versión 5.0.0:

Install-Package EntityFramework -Version 5.0.0

# Crear la clase de Contexto

En el Proyecto creado:

🡪Agregue una clase BetaGamerContext en la carpeta Models.

🡪Haga que herede de DbContext

🡪Cree las propiedades de tipo DbSet para los tipos Juego y Comentario:

public class BetaGamerContext:DbContext{

public DbSet<Juego> Juegos { get; set; }

public DbSet<Comentario> Comentarios { get; set; }

}

🡪Sobrescriba el constructor de la base parametrizando el nombre de la clave del connectionString en el web.config.

public BetaGamerContext():base("con"){}

**Nota**: Recuerde incluir en la sección ConnectionStrings la conexión ***con***:

<connectionStrings>

<add name="con" connectionString="Data Source=(local); Initial Catalog=BetaGamers; Integrated Security=SSPI;" providerName="System.Data.SqlClient"/>

</connectionStrings>

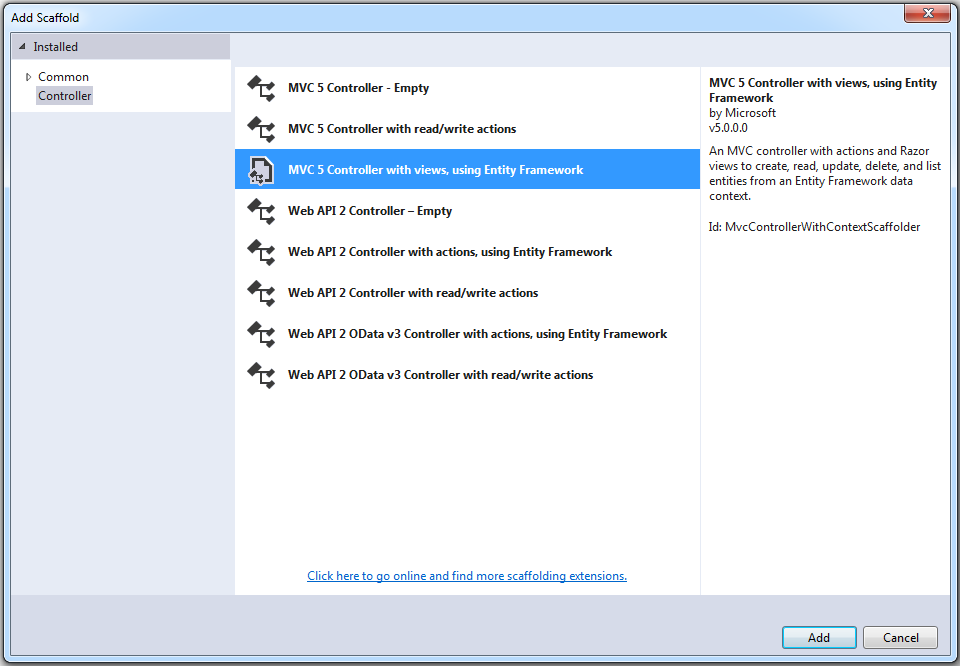
# Crear el Controller para la clase JUEGO

Vamos a crear un controlador que usa BetaGamerContext para traer/guardar datos.

Para la clase Juego vamos a crear una vista que muestre la lista de todos los juegos disponibles, y de ahí llamar a otra que permita crear nuevos juegos.

Compile el proyecto. En el explorador de Soluciones:

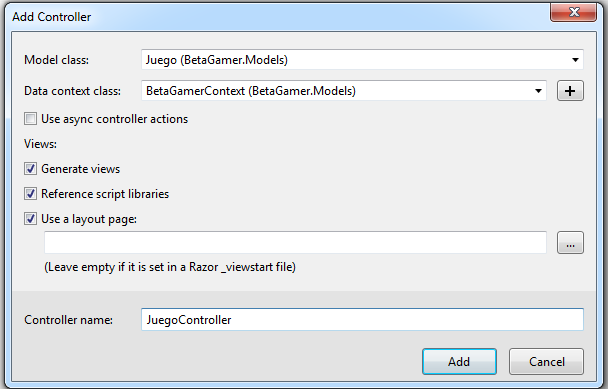
🡪click derecho sobre la carpeta Controllers🡪Add🡪Controller🡪Plantilla: Controlador MCV 5 con vistas que usa EF. 🡪Agregar



🡪Clase de Modelo: Juego

🡪Clase de contexto de datos: BetaGamerContext

🡪Nombre del controlador: JuegoController



🡪Agregar  
  
Se crea JuegoController con métodos ActioResult para Index, Details, Create, Edit and Delete. El Index retornará la lista de Juegos en una vista.

# vistas para el controller

Navegue a la carpeta Views. Note que se creó una carpeta Juego y dentro una vista para cada acción.

# Ruteo.

Las instrucciones de ruteo por defecto están en la Carpeta App\_Start, en la clase RouteConfig.cs.

public static void RegisterRoutes(RouteCollection routes)

{

routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");

routes.MapRoute(

name: "Default",

url: "{controller}/{action}/{id}",

defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }

);

}

Nota: Si quisiéramos cambiar la página de inicio podríamos cambiar Home por Juego y al ejecutar la aplicación invocaría la acción JuegoController.Index la que recuperará la lista de Juegos y los retornará en la vista por defecto.

# Antes de ejecutar

Abra la master page \_Layout en la carpeta Shared

Encuentre el Menú y agregue un ítem para invocar a lista de Juegos.

<li>@Html.ActionLink("Juegos", "Index", "Juego")</li>

# Probar la aplicación

Haga click en el link Create, este invocará el método Create del controlador y retornará la vista asociada. Cargue datos para el nuevo Juego y presione el botón Create. El control es pasado al controlador y se llama al segundo método Create al que la vista envía un objeto Juego creado.

Ahora podremos ver el Nuevo Juego en la lista desplegada en el Index.

Pruebe las restantes acciones.