(T4)討論 MvcConventions(命名規則)。討論 AdoNet、BusinessLayer 的 Insert
CourseGUID: 8503b39c-5887-4634-8291-facfb3117924
==
(T4)討論 MvcConventions(命名規則)。討論 AdoNet、BusinessLayer 的 Insert
==

==

0. Summary

In this tutorial, we will discuss
* MvcConventions
* AdoDotNet
* BusinessLayer
* InsertData
This is continuous with the previous tutorial.
Please ensure you finish the previous tutorial before you continue this tutorial.

Tutorial 4: 完全手寫 ADO.NET 連接資料庫以及新增/更新/移除資料
大部分市面上英文版的 ASP.NET MVC 課程都是一面倒只介紹 Entity Framework。可是我們知道目前其實還是有許多公司使用 Asp.Net Web Form Application 搭配 ADO.NET。那如果該公司想要更新技術使用 ASP.NET MVC 搭配 Entity Framework。那你卻因為市面上大部分的 MVC 課程沒介紹連接 ADO.NET 因而 GG，那是多冤枉啊。所以本課程也介紹如何讓 MVC 連接 ADO.NET。使用舊技術的公司可以先在過渡期繼續使用 ADO.NET 搭配 MVC，不必馬上就使用 Entity Framework 造成過渡期短時間花費過多學習程本。 你只有透過完全手寫，才能完善你的理解。
==
補充 1
viewBag vs viewData
-->
A
1.

ViewBag is just a wrapper around the ViewData.
2.
ViewData also works with .net framework 3.5 and above
3.
ViewBag only works with .net framework 4.0 and above
4.
ViewData :
Type Conversion code is required while enumerating
5.
ViewBag:
In depth, ViewBag is used dynamic, so there is no need to type conversion while enumerating.
原本只有 ViewData，後來微軟生一個 ViewBag 出來應該不是無聊沒事做。所以 ViewData 和 ViewBag，我大部分用 ViewBag。
===

1. MVC conventions

In MVC conventions,
1. Controllers must have the word "Controller" as the suffix and must extend "IController" interface.
2. A view must remain under "Views" folder.
3. If the view is for GamerController, then the view must remain under "Views/Gamer" folder.
4. In the "HomeController", when "Index" action "return View()", it will search the following files in order.
   4.1. ~/Views/Home/Index.aspx
   4.2. ~/Views/Home/Index.ascx
   4.3. ~/Views/Shared/Index.aspx
   4.4. ~/Views/Shared/Index.ascx
   4.5. ~/Views/Home/Index.cshtml
   4.6. ~/Views/Home/Index.vbhtml
   4.7. ~/Views/Shared/Index.cshtml
   4.8. ~/Views/Shared/Index.vbhtml
5. By MVC convention, MVC will look for the view in the following locations
   5.1. Views/ControllerName
   5.2. Views/Shared
6. The extension name of view can be cshtml, vbhtml, aspx, or ascx.
7. Models can be any where, even can be in other project. However, it is better to put it in "Models" folder.
8. You may put Models in another project as business layer.

1.1. Controllers/HomeController.cs

```
using System.Collections.Generic;
using System.Web.Mvc;
namespace OnlineGame.Web.Controllers
{
    public class HomeController : Controller
     {
        //// GET: Home
        //public string Index()
        //{
```

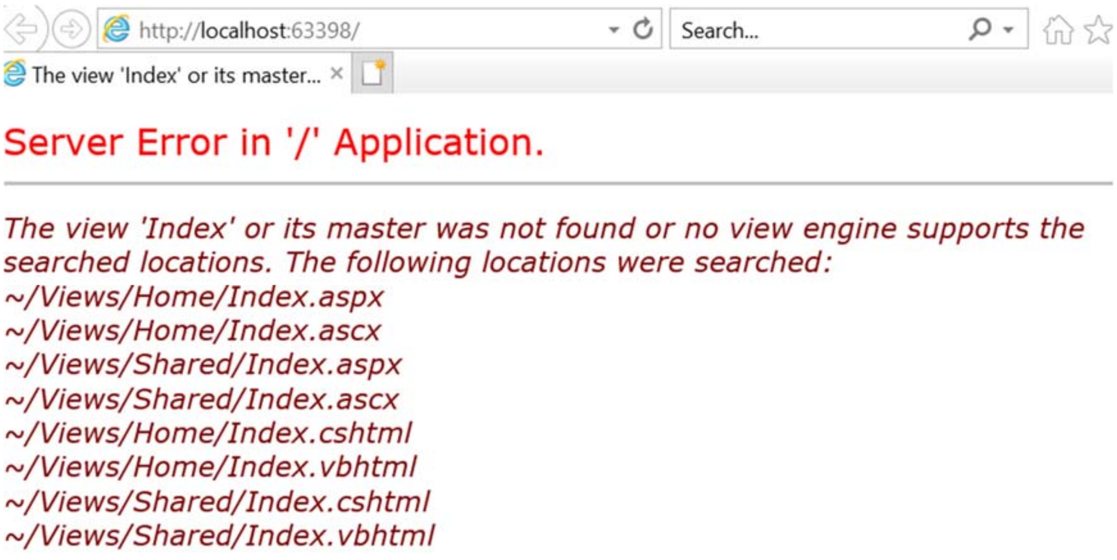
```
//    return "Hello";
//}
////http://localhost/OnlineGame.Web/home/index/aa?name=bbb
////http://localhost/OnlineGame.Web/home/index/aa?name2=bbb
//public string Index(string id)
//{
//    string queryString = Request.QueryString["name"];
//    return $"Hey, Id={id} , name={queryString}";
//}
////http://localhost/OnlineGame.Web/home/index/aa?name=bbb
////http://localhost/OnlineGame.Web/home/index/aa?name2=bbb
//public string Index(string id, string name)
//{
//    // return string.Format("Hey, Id ={0} , name ={1}", id, name);
//    return $"Hey, Id ={id} , name ={name}";
//}
//public List<string> Index()
//{
//    return new List<string>
//    {
//        "Name01",
//        "Name02",
//        "Name03"
//    };
//    // Return System.Collections.Generic.List`1[System.String]
//    // This is Wrong.
//}
//public ActionResult Index()
//{
//    return View();
//}
//public ActionResult Index()
//{
//    ViewBag.Names = new List<string>
//        {
//            "Name01",
//            "Name02",
//            "Name03"
//        };
//    return View();
//}
public ActionResult Index()
{
    ////1.
    //ViewBag.Names = new List<string>
    //{
    //    "ViewBag.Names01",
    //    "ViewBag.Names02",
    //    "ViewBag.Names03"
    //};
    ////2.
    //ViewData["Names"] = new List<string>
    //{
            //    "ViewData[\"Names\"]01",
            //    "ViewData[\"Names\"]02",
            //    "ViewData[\"Names\"]03"
            //};
            ////3.
            //ViewBag.Names = new List<string>
            //{
            //    "ViewBag.Names01",
            //    "ViewBag.Names02",
            //    "ViewBag.Names03"
            //};
            //ViewData["Names"] = new List<string>
            //{
            //    "ViewData[\"Names\"]01",
            //    "ViewData[\"Names\"]02",
            //    "ViewData[\"Names\"]03"
            //};
            //4.
            ViewBag.Names = new List<string>
            {
                "ViewBag.Names01",
                "ViewBag.Names02",
                "ViewBag.Names03"
            };
            ViewData["Names2"] = new List<string>
            {
                "ViewData[\"Names\"]01",
                "ViewData[\"Names\"]02",
                "ViewData[\"Names\"]03"
            };
            return View();
        }
        public string GetStringA()
        {
            return "AAAAAA";
        }
    }
}
/*
1.
When we try to return a list of data,
E.g.
return new List<string>
{
    "Name01",
    "Name02",
    "Name03"
};
Then, it will only display the data type of the variable
E.g.
System.Collections.Generic.List`1[System.String]
This is not what we want,
thus, we need a view to display the data in the format we want.
2.
//public ActionResult Index()
//{
//    return View();
//}
ViewResult extend ViewResultBase
```

```
ViewResultBase extend ActionResult.
Thus, you can return View()
3.
In Home/HomeController.cs
//ViewBag.Names = new List<string>
//{
//    "ViewBag.Names01",
//    "ViewBag.Names02",
//    "ViewBag.Names03"
//};
//ViewData["Names2"] = new List<string>
//{
//    "ViewData[\"Names\"]01",
//    "ViewData[\"Names\"]02",
//    "ViewData[\"Names\"]03"
//};
In Views/HomeIndex.cshtml
//@foreach (string strNames1 in ViewBag.Names)
//{
//    <li>@strNames1</li>
//}
//<br/>
//<br/>
//@foreach (string strNames2 in (List<string>) ViewData["Names2"])
//{
//    <li>@strNames2</li>
//}
Both ViewData and ViewBag can pass values from Controller to View.
Both ViewData and ViewBag allow an object to have properties dynamically added to it.
Because of dynamic feature,
both ViewData and ViewBag does not provide compile time error checking.
Thus, it is very easy to get Null Reference Error
if miss misspells the property name or key name.
*/
```

1.2. Views/Home/Index.cshtml

Now you delete   Views/Home/Index.cshtml
You will see the following error message.

The view 'Index' or its master was not found or no view engine supports the searched locations. The following locations were searched:
~/Views/Home/Index.aspx
~/Views/Home/Index.ascx
~/Views/Shared/Index.aspx
~/Views/Shared/Index.ascx
~/Views/Home/Index.cshtml
~/Views/Home/Index.vbhtml
~/Views/Shared/Index.cshtml
~/Views/Shared/Index.vbhtml



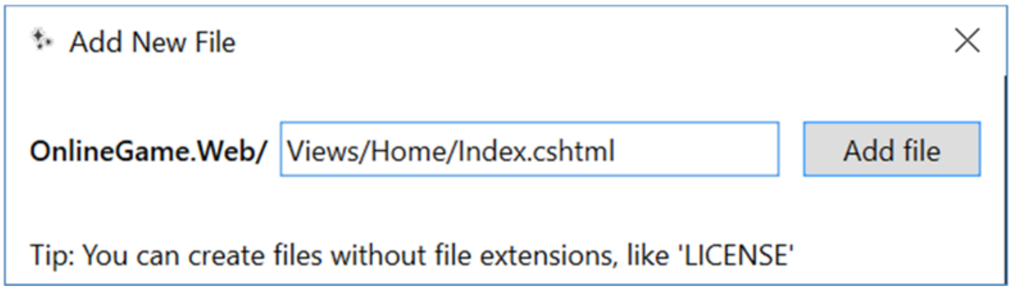
Server Error in '/' Application.

The view 'Index' or its master was not found or no view engine supports the searched locations. The following locations were searched:
~/Views/Home/Index.aspx
~/Views/Home/Index.ascx
~/Views/Shared/Index.aspx
~/Views/Shared/Index.ascx
~/Views/Home/Index.cshtml
~/Views/Home/Index.vbhtml
~/Views/Shared/Index.cshtml
~/Views/Shared/Index.vbhtml

Because Add New File **(extension and update)**

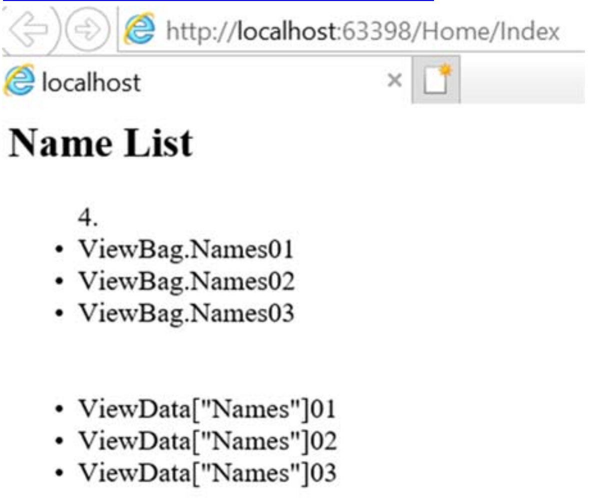
press **Shift+F2**

Views/Home/Index.cshtml


Add New File
OnlineGame.Web/
Views/Home/Index.cshtml
Add file
Tip: You can create files without file extensions, like 'LICENSE'

```
@model dynamic
@{
    ViewBag.Title = "title";
}
<h2>Name List</h2>
<ul>
    @*
        1.
        @foreach (string strNames1 in ViewBag.Names)
        {
            <li>@strNames1</li>
        }
    *@
    @*
        2.
        @foreach (string strNames2 in (List<string>)ViewData["Names"])
        {
            <li>@strNames2</li>
        }
    *@
    @*
        3.
        @foreach (string strNames1 in ViewBag.Names)
        {
            <li>@strNames1</li>
        }
        <br />
        <br />
        @foreach (string strNames2 in (List<string>)ViewData["Names"])
        {
            <li>@strNames2</li>
        }
    *@
    4.
    @foreach (string strNames1 in ViewBag.Names)
    {
        <li>@strNames1</li>
    }
    <br />
    <br />
    @foreach (string strNames2 in (List<string>)ViewData["Names2"])
    {
        <li>@strNames2</li>
    }
</ul>
```

http://localhost:63398/Home/Index


http://localhost:63398/Home/Index

localhost

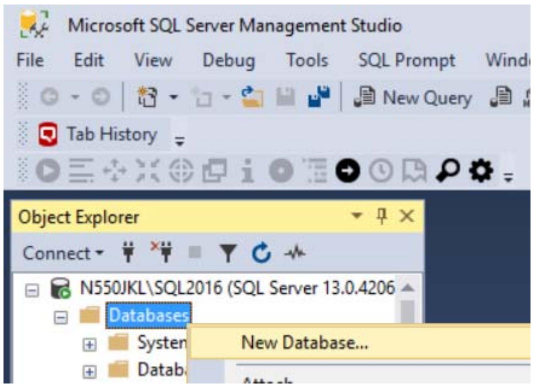
Name List

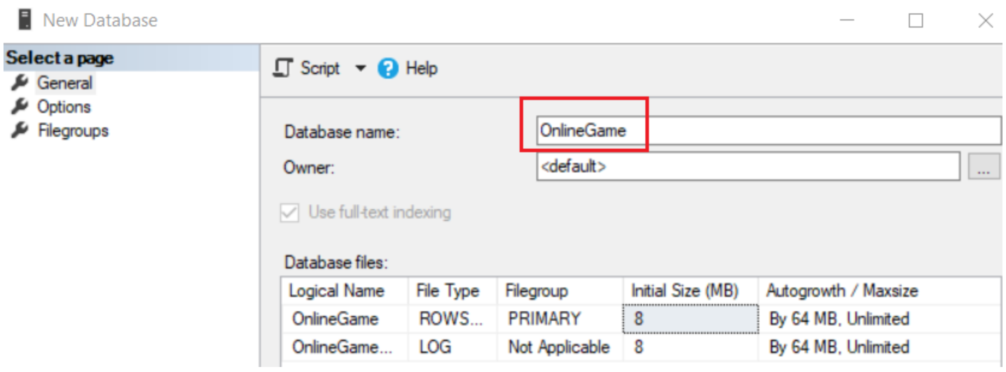
4.
- ViewBag.Names01
- ViewBag.Names02
- ViewBag.Names03

- ViewData["Names"]01
- ViewData["Names"]02
- ViewData["Names"]03

2. OnlineGame DB

In SQL server Management Studio (SSMS)
Database --> Right Click --> New Database -->
In General Tab -->
Name: **OnlineGame**
In options Tab --> Recovery model : **Simple**


Microsoft SQL Server Management Studio
File Edit View Debug Tools SQL Prompt Wind
New Query
Tab History
Object Explorer
Connect
N550JKL\SQL2016 (SQL Server 13.0.4206
Databases
Systen
Datab
New Database...


New Database

Select a page
- General
- Options
- Filegroups

Script ▾ Help

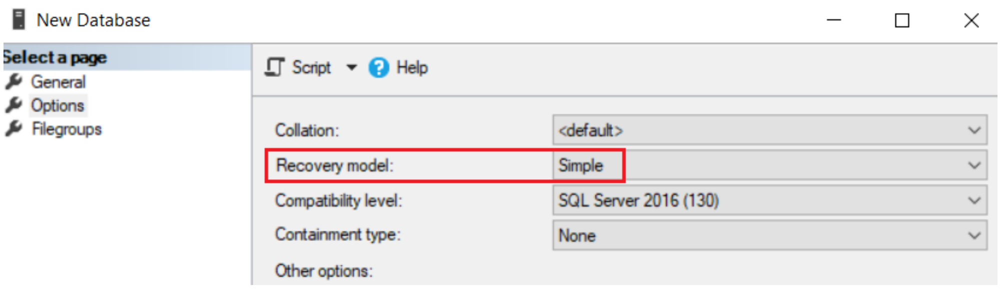
Database name: OnlineGame

Owner: <default>

☑ Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize
OnlineGame	ROWS...	PRIMARY	8	By 64 MB, Unlimited
OnlineGame...	LOG	Not Applicable	8	By 64 MB, Unlimited


New Database

Select a page
- General
- Options
- Filegroups

Script ▾ Help

Collation:	<default>
Recovery model:	Simple
Compatibility level:	SQL Server 2016 (130)
Containment type:	None

Other options:

```
--1. Drop if it exists
--Drop Table if it exists.
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Gamer' ) )
    BEGIN
        TRUNCATE TABLE Gamer;
        DROP TABLE Gamer;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Team' ) )
    BEGIN
        TRUNCATE TABLE Team;
        DROP TABLE Team;
    END;
GO -- Run the previous command and begins new batch
--Drop Stored Procedure if it exists.
--IF OBJECT_ID('spSearchGamer') IS NOT NULL
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.ROUTINES
              WHERE     ROUTINE_TYPE = 'PROCEDURE'
                        AND LEFT(ROUTINE_NAME, 3) NOT IN ( 'sp_', 'xp_', 'ms_' )
                        AND SPECIFIC_NAME = 'spGetGamers' ) )
    BEGIN
        DROP PROCEDURE spGetGamers;
    END;
GO -- Run the previous command and begins new batch
--IF OBJECT_ID('spSearchGamer') IS NOT NULL
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.ROUTINES
```

```
                WHERE      ROUTINE_TYPE = 'PROCEDURE'
                           AND LEFT(ROUTINE_NAME, 3) NOT IN ( 'sp_', 'xp_', 'ms_' )
                           AND SPECIFIC_NAME = 'spAddGamer' ) )
    BEGIN
        DROP PROCEDURE spAddGamer;
    END;
GO -- Run the previous command and begins new batch
--2. Create Table
CREATE TABLE Team
    (
      Id INT PRIMARY KEY
             IDENTITY(1, 1)
             NOT NULL ,
      [Name] NVARCHAR(100) NULL
    );
GO -- Run the previous command and begins new batch
CREATE TABLE Gamer
    (
      Id INT PRIMARY KEY
             IDENTITY(1, 1)
             NOT NULL ,
      [Name] NVARCHAR(100) NULL ,
      Gender NVARCHAR(10) NULL ,
      City NVARCHAR(50) NULL ,
      DateOfBirth DATETIME NULL ,
      TeamId INT FOREIGN KEY REFERENCES Team ( Id )
    );
GO -- Run the previous command and begins new batch
--3. Insert Data
INSERT  Team
VALUES  ( N'Team1' );
INSERT  Team
VALUES  ( N'Team2' );
INSERT  Team
VALUES  ( N'Team3' );
INSERT  Gamer
VALUES  ( N'Name01 ABB', N'Male', N'City01', '1979/4/28', 1 );
INSERT  Gamer
VALUES  ( N'Name02 CDDE', N'Female', N'City03', '1981/7/24', 2 );
INSERT  Gamer
VALUES  ( N'Name03 FIJK', N'Female', N'City01', '1984/12/5', 3 );
INSERT  Gamer
VALUES  ( N'Name04 LMOPPQ', N'Male', N'City02', '1983/5/29', 1 );
INSERT  Gamer
VALUES  ( N'Name05 QRSTT', N'Male', N'City01', '1979/6/20', 3 );
INSERT  Gamer
VALUES  ( N'Name06 TUVVX', N'Female', N'City03', '1984/5/15', 3 );
INSERT  Gamer
VALUES  ( N'Name07 XYZZXX', N'Female', N'City01', '1986/4/29', 2 );
INSERT  Gamer
VALUES  ( N'Name08 ABBCDE', N'Male', N'City02', '1985/7/28', 1 );
INSERT  Gamer
VALUES  ( N'Name09 QRSTTUVXX', N'Male', N'City02', '1983/4/16', 1 );
GO -- Run the previous command and begins new batch
--4. SP
CREATE PROCEDURE spGetGamers
```

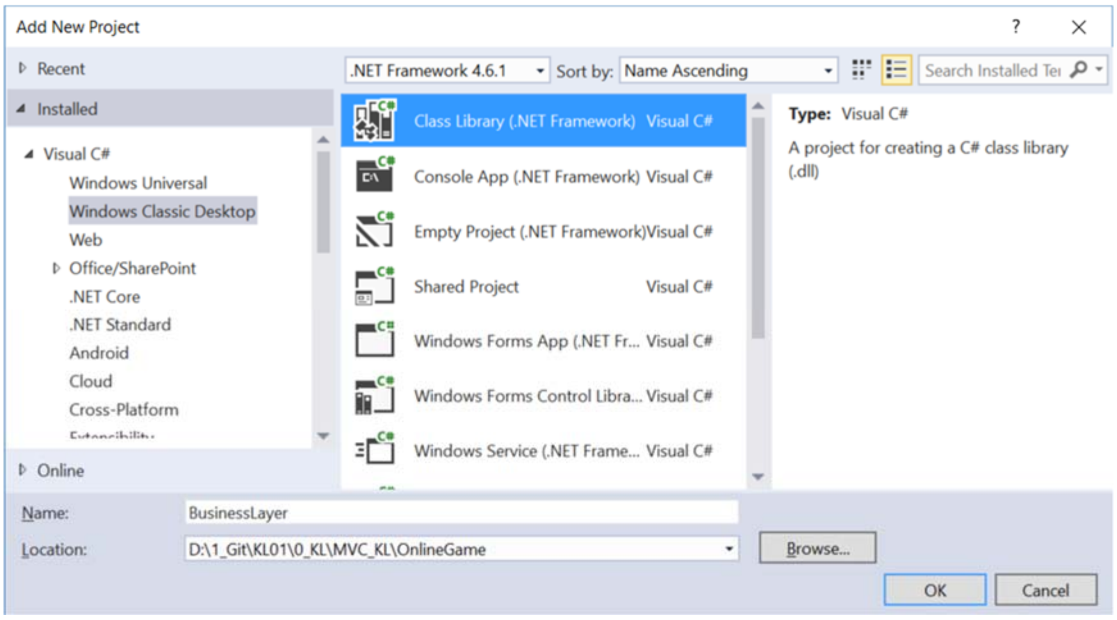
```
AS
    BEGIN
        SELECT  *
        FROM    Gamer;
    END;
GO -- Run the previous command and begins new batch
CREATE PROCEDURE spAddGamer
    (
      @Name NVARCHAR(50) ,
      @Gender NVARCHAR(10) ,
      @City NVARCHAR(50) ,
      @DateOfBirth DateTime ,
      @TeamId INT
    )
AS
    BEGIN
        INSERT  INTO Gamer
        VALUES  ( @Name, @Gender, @City, @DateOfBirth, @TeamId );
    END;
GO -- Run the previous command and begins new batch

--EXEC spGetGamers
--GO -- Run the previous command and begins new batch
```

3. Add Class Library - BusinessLayer, ADO.NET

In this BusinessLayer, we will use Ado.Net to connect database.

Solution Name --> Add --> New Project -->
Visual C# --> Windows Classic Desktop --> Class Library (.NET Framework)
-->
Name:
BusinessLayer


Add New Project

.NET Framework 4.6.1 | Sort by: Name Ascending | Search Installed Tei

Recent
Installed
Visual C#
Windows Universal
Windows Classic Desktop
Web
Office/SharePoint
.NET Core
.NET Standard
Android
Cloud
Cross-Platform
Online

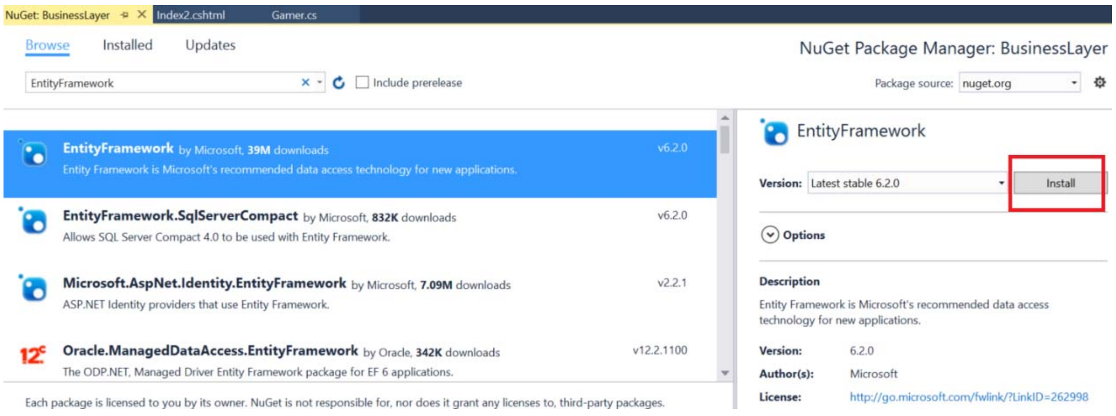
Class Library (.NET Framework)	Visual C#
Console App (.NET Framework)	Visual C#
Empty Project (.NET Framework)	Visual C#
Shared Project	Visual C#
Windows Forms App (.NET Fr...	Visual C#
Windows Forms Control Libra...	Visual C#
Windows Service (.NET Frame...	Visual C#

Type: Visual C#
A project for creating a C# class library (.dll)

Name: BusinessLayer
Location: D:\1_Git\KL01\0_KL\MVC_KL\OnlineGame

3.1. Add Reference

Add "EntityFramework"



NuGet Package Manager: BusinessLayer

EntityFramework | Include prerelease | Package source: nuget.org

EntityFramework by Microsoft, **39M** downloads v6.2.0
Entity Framework is Microsoft's recommended data access technology for new applications.

EntityFramework.SqlServerCompact by Microsoft, **832K** downloads v6.2.0
Allows SQL Server Compact 4.0 to be used with Entity Framework.

Microsoft.AspNet.Identity.EntityFramework by Microsoft, **7.09M** downloads v2.2.1
ASP.NET Identity providers that use Entity Framework.

Oracle.ManagedDataAccess.EntityFramework by Oracle, **342K** downloads v12.2.1100
The ODP.NET, Managed Driver Entity Framework package for EF 6 applications.

Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages.

EntityFramework

Version: Latest stable 6.2.0 | Install

Options

Description
Entity Framework is Microsoft's recommended data access technology for new applications.

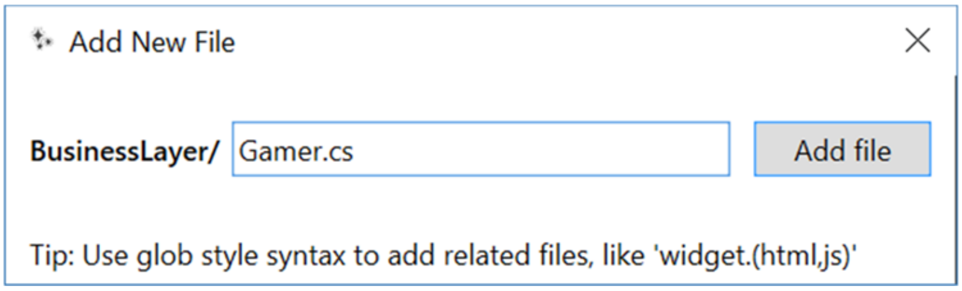
Version: 6.2.0
Author(s): Microsoft
License: http://go.microsoft.com/fwlink/?LinkID=262998

3.2. BusinessLayer/Gamer.cs

Because Add New File **(extension and update)**

press **Shift+F2**

BusinessLayer/Gamer.cs



Add New File

BusinessLayer/ Gamer.cs | Add file

Tip: Use glob style syntax to add related files, like 'widget.(html,js)'

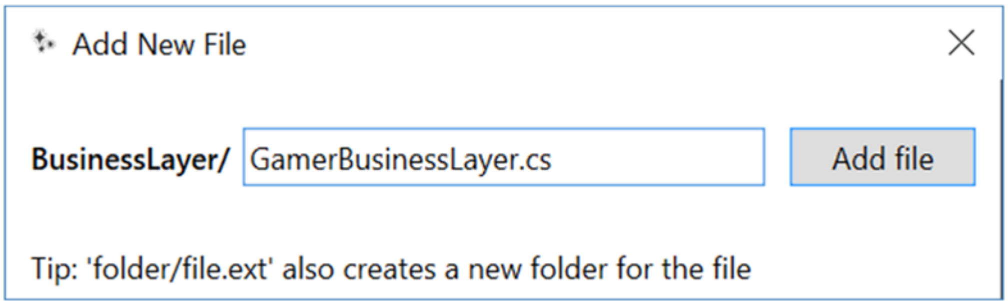
```
using System;
using System.ComponentModel.DataAnnotations;
namespace BusinessLayer
{
    public class Gamer
    {
        public int Id { get; set; }
         [Required]
        public string Name { get; set; }
         [Required]
        public string Gender { get; set; }
         [Required]
        public string City { get; set; }
         [Required]
        public DateTime DateOfBirth { get; set; }
         [Required]
        public int TeamId { get; set; }
    }
}
```

3.3. BusinessLayer/GamerBusinessLayer.cs (ADO.NET)

Because Add New File **(extension and update)**

press **Shift+F2**

BusinessLayer/GamerBusinessLayer.cs



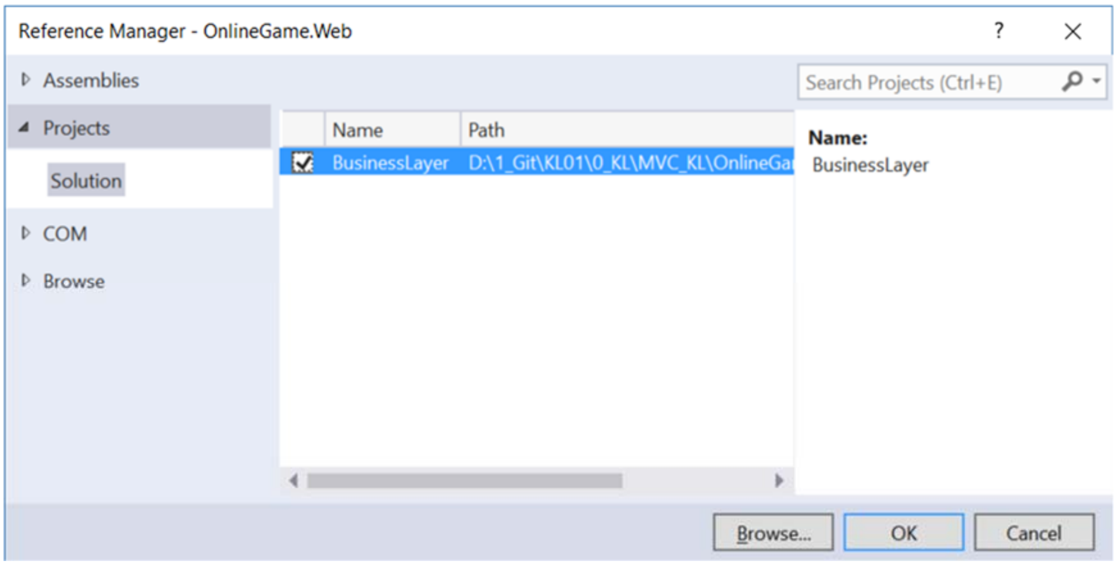
Tip: 'folder/file.ext' also creates a new folder for the file

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
namespace BusinessLayer
{
    public class GamerBusinessLayer
    {
        public IEnumerable<Gamer> Gamers
        {
            get
            {
                string connectionString =
                    ConfigurationManager.ConnectionStrings["OnlineGameContext"].ConnectionString;
                List<Gamer> gamers = new List<Gamer>();
                using (SqlConnection con = new SqlConnection(connectionString))
                {
                    SqlCommand cmd = new SqlCommand("spGetGamers", con);
cmd.CommandType = CommandType.StoredProcedure;
                    con.Open();
                    SqlDataReader rdr = cmd.ExecuteReader();
                    while (rdr.Read())
                    {
                        Gamer gamer = new Gamer();
                        gamer.Id = Convert.ToInt32(rdr["Id"]);
                        gamer.Name = rdr["Name"].ToString();
                        gamer.Gender = rdr["Gender"].ToString();
                        gamer.City = rdr["City"].ToString();
                        gamer.DateOfBirth = Convert.ToDateTime(rdr["DateOfBirth"]);
                        gamer.TeamId = Convert.ToInt32(rdr["TeamId"]);
                        gamers.Add(gamer);
                    }
                }
                return gamers;
            }
        }
        public void AddGamer(Gamer gamer)
        {
            string connectionString =
            ConfigurationManager.ConnectionStrings["OnlineGameContext"].ConnectionString;
            using (SqlConnection con = new SqlConnection(connectionString))
            {
                SqlCommand cmd = new SqlCommand("spAddGamer", con)
                {
                    CommandType = CommandType.StoredProcedure
                };
                SqlParameter sqlParamName = new SqlParameter
                {
                    ParameterName = "@Name",
                    Value = gamer.Name
                };
                cmd.Parameters.Add(sqlParamName);
                SqlParameter sqlParamGender = new SqlParameter
                {
                    ParameterName = "@Gender",
                    Value = gamer.Gender
                };
                cmd.Parameters.Add(sqlParamGender);
                SqlParameter sqlParamCity = new SqlParameter
                {
                    ParameterName = "@City",
                    Value = gamer.City
                };
                cmd.Parameters.Add(sqlParamCity);
                SqlParameter sqlParamDateOfBirth = new SqlParameter
                {
                    ParameterName = "@DateOfBirth",
                    Value = gamer.DateOfBirth
                };
                cmd.Parameters.Add(sqlParamDateOfBirth);
                SqlParameter sqlParamTeamId = new SqlParameter
                {
                    ParameterName = "@TeamId",
                    Value = gamer.TeamId
                };
                cmd.Parameters.Add(sqlParamTeamId);
                con.Open();
                cmd.ExecuteNonQuery();
            }
        }
    }
}
```

4. OnlineGame.Web/Controllers/GamerController.cs

4.1. Add Reference

Add Reference "**BusinessLayer**" into "**OnlineGame.Web**"


Reference Manager - OnlineGame.Web

	Name	Path
✓	BusinessLayer	D:\1_Git\KL01\0_KL\MVC_KL\OnlineGa

Name:
BusinessLayer

4.2. OnlineGame.Web/Controllers/GamerController.cs

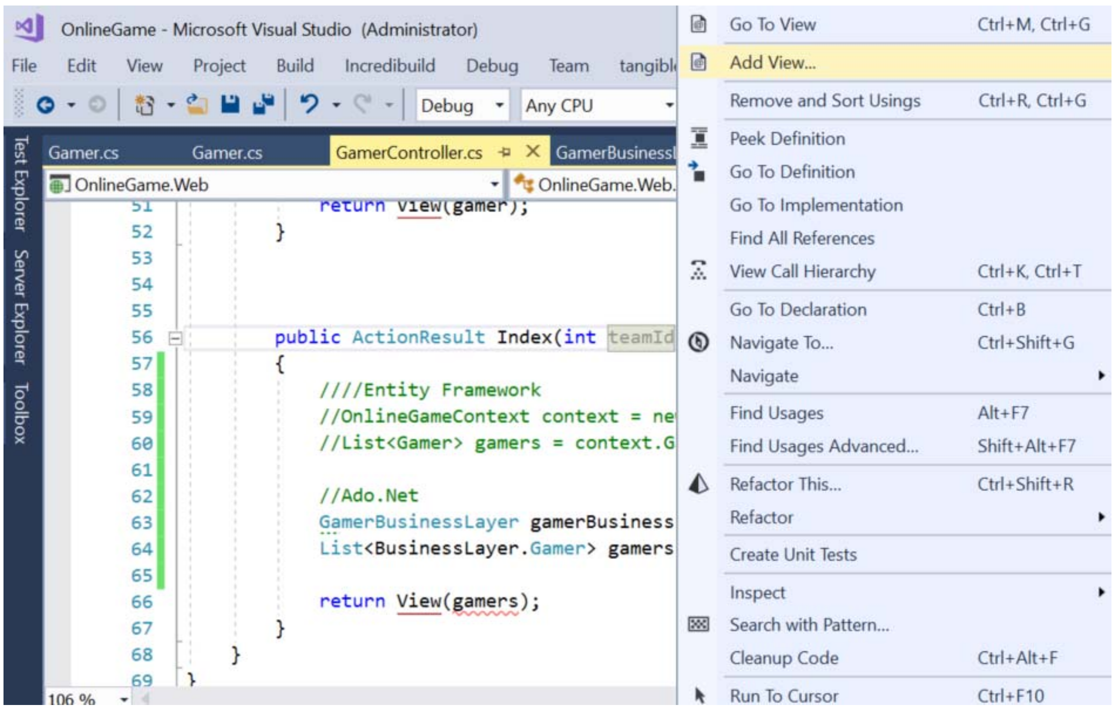
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web.Mvc;
using BusinessLayer;
using OnlineGame.Web.Data;
using Gamer = OnlineGame.Web.Models.Gamer;
namespace OnlineGame.Web.Controllers
{
    public class GamerController : Controller
    {
        // http://localhost/OnlineGame.Web/Gamer/Details
        //public ActionResult Details()
        //{
        //    var gamer = new Gamer
        //    {
        //        Id = 1,
        //        Name = "Name1",
        //        Gender = "Male",
        //        City = "City1"
        //    };
        //    return View(gamer);
        //}
        // http://localhost/OnlineGame.Web/Gamer/Details
        // http://localhost/OnlineGame.Web/Gamer/Details/1
        // http://localhost/OnlineGame.Web/Gamer/Details/2
        // http://localhost/OnlineGame.Web/Gamer/Details/3
        // http://localhost/OnlineGame.Web/Gamer/Details/4
        public ActionResult Details(int id = 0)
        {
            var onlineGameContext = new OnlineGameContext();
            Gamer gamer;
            if (id == 0)
            {
                gamer = new Gamer
                {
                    Id = 0,
                    Name = "Name0",
                    Gender = "NULL",
                    City = "NULL"
                };
                // or you may throw exception here.
            }
            else
            {
                gamer = onlineGameContext.Gamers.Single(p => p.Id == id);
                //Throws exception if can not find the single entity
            }
            return View(gamer);
        }
        public ActionResult Index(int teamId)
        {
            //Entity Framework
            OnlineGameContext context = new OnlineGameContext();
            List<Gamer> gamers = context.Gamers.Where(gamer => gamer.TeamId == teamId).ToList();
            return View(gamers);
        }
        public ActionResult Index2()
        {
            //Ado.Net
            GamerBusinessLayer gamerBusinessLayer = new GamerBusinessLayer();
            List<BusinessLayer.Gamer> gamers = gamerBusinessLayer.Gamers.ToList();
            return View(gamers);
        }
        //[HttpGet] attribute means it only respond to the "GET" request.
        [HttpGet]
        public ActionResult Create()
        {
            return View();
        }
        // 1. Retrieve form data using FormCollection
        [HttpPost]
        public ActionResult Create(FormCollection formCollection)
        {
            ////FormCollection implement C# indexer.
            ////See each key and value of formCollection.
            //foreach (string key in formCollection.AllKeys)
            //{
            //    Response.Write($"key=={key}, {formCollection[key]}, <br/>");
            //}
            int teamId;
            BusinessLayer.Gamer gamer = new BusinessLayer.Gamer
            {
                Name = formCollection["Name"],
                Gender = formCollection["Gender"],
                City = formCollection["City"],
                DateOfBirth = Convert.ToDateTime(formCollection["DateOfBirth"]),
                TeamId = int.TryParse(formCollection["TeamId"], out teamId) ? teamId : 0
            };
            GamerBusinessLayer gamerBusinessLayer =
                new GamerBusinessLayer();
            gamerBusinessLayer.AddGamer(gamer);
            return RedirectToAction("Index2");
        }
        // 2. Retrieve form data using name attribute of input tag from cshtml
        [HttpPost]
        public ActionResult Create2(string name, string gender, string city, DateTime dateOfBirth, int teamId)
        {
            BusinessLayer.Gamer gamer = new BusinessLayer.Gamer
            {
                Name = name,
                Gender = gender,
                City = city,
                DateOfBirth = dateOfBirth,
                TeamId = teamId
            };
            GamerBusinessLayer gamerBusinessLayer =
                new GamerBusinessLayer();
            gamerBusinessLayer.AddGamer(gamer);
            return RedirectToAction("Index2");
        }
        // 3. Retrieve form data using model binding
        [HttpPost]
        public ActionResult Create3(BusinessLayer.Gamer gamer)
        {
            //if any of input is not valid.
            if (!ModelState.IsValid)
            {
                return View("Create");
                //Go to Create.cshtml,
                //so users can correct their input value.
            }
            GamerBusinessLayer gamerBusinessLayer =
                new GamerBusinessLayer();
            gamerBusinessLayer.AddGamer(gamer);
            return RedirectToAction("Index2");
        }
        // 4. Retrieve form data using model binding by UpdateModel() or TryUpdateModel()
        [HttpPost]
        [ActionName("Create4")]
        public ActionResult Create_Post()
        {
            //if any of input is not valid.
            if (!ModelState.IsValid)
            {
                return View("Create");
                //Go to Create.cshtml,
                //so users can correct their input value.
            }
            GamerBusinessLayer gamerBusinessLayer =
                new GamerBusinessLayer();
            BusinessLayer.Gamer gamer = new BusinessLayer.Gamer();
            //UpdateModel<BusinessLayer.Gamer>(gamer);
            //UpdateModel(gamer);
            TryUpdateModel(gamer);
            //1.
            // UpdateModel() and TryUpdateModel() inspects all the HttpRequest inputs
            // such as posted Form data, QueryString,
            // Cookies and Server variables and populate the gamer object.
            gamerBusinessLayer.AddGamer(gamer);
            return RedirectToAction("Index2");
        }
    }
}
/*
1.
//var onlineGameContext = new OnlineGameContext();
//Gamer gamer = onlineGameContext.Gamers.Single(p => p.Id == id);
When user request, EntityFramework will request the data from the database
and sotre its data into a temp place called DBSet.
onlineGameContext.Gamers is a DBSet which is kind of temp place to store the Gamer Table Data.
We use LINQ to map the Gamer Table Column id to Gamer Model property, id.
Thus, we can get the gamer entity from Gamer Table by its id.
Then store gamer entity data into Gamer Model object.
Thus, each Gamer Model object is a temp place to store each Gamer Table entity from the database.
Then we pass the Gamer Model object as the ViewModel,
Thus, the Details.cshtml view can use the values from Gamer Model object
which is actually the temp place to store Gamer Table entity data.
-------------------------------
2.
//[HttpGet]
//public ActionResult Create()
The GET request will direct to Views/Gamer/Create.cshtml.


-------------------------------
3.
//[HttpPost]
-----------------
3.1.
//[HttpPost]
//public ActionResult Create(FormCollection formCollection)
Retrieve form data using FormCollection.
The key is the name attribute of input or select tag from cshtml.
-----------------
3.2.
//[HttpPost]
//public ActionResult Create2(string name, string gender, string city, DateTime dateOfBirth, int teamId)
Retrieve form data using name attribute of input tag from cshtml.
3.2.1.
string name is from
//<input class="form-control text-box single-line" id="Name" name="Name" type="text" value="">
3.2.2.
string gender is from
//<select id="Gender" name="Gender">...</select>
3.2.3.
string city is from
<input class="form-control text-box single-line" id="City" name="City" type="text" value="">
3.2.4.
DateTime dateOfBirth is from
//<input class="form-control text-box single-line" data-val="true" data-val-date="The field DateOfBirth
must be a date." data-val-required="The DateOfBirth field is required." id="DateOfBirth"
name="DateOfBirth" type="datetime" value="">
3.2.5.
int teamId is from
//<input class="form-control text-box single-line" data-val="true" data-val-number="The field TeamId must
be a number." data-val-required="The TeamId field is required." id="TeamId" name="TeamId" type="number"
value="">
-----------------
3.3.
//[HttpPost]
//public ActionResult Create3(BusinessLayer.Gamer gamer)
If the view has a lot of input,
then the previous two ways is not a good idea.
It is always better to retrieve form data using model binding.
The model of the cshtml is BusinessLayer.Gamer,
so we can pass the model object into HttpPost action.
The property value of model object will contain the value
```

```
from input or select tag from cshtml based on name attribute.
-----------------
3.4.
//[HttpPost]
//[ActionName("Create4")]
//public ActionResult Create_Post()
Retrieve form data using model binding by UpdateModel() or TryUpdateModel()
...
//BusinessLayer.Gamer gamer = new BusinessLayer.Gamer();
////UpdateModel<BusinessLayer.Gamer>(gamer);
////UpdateModel(gamer);
//TryUpdateModel(gamer);
3.4.1.
UpdateModel() and TryUpdateModel() inspects all the HttpRequest inputs
such as posted Form data, QueryString,
Cookies and Server variables and populate the gamer object.
3.4.2.
UpdateModel() throws an exception if validation fails.
TryUpdateModel() will never throw an exception and
return false if validation fails.
*/
```

4.3. OnlineGame.Web/Views/Gamer/Index2.cshtml

Delete OnlineGame.Web/Views/Gamer/Index.cshtml
before you continue this tutorial.

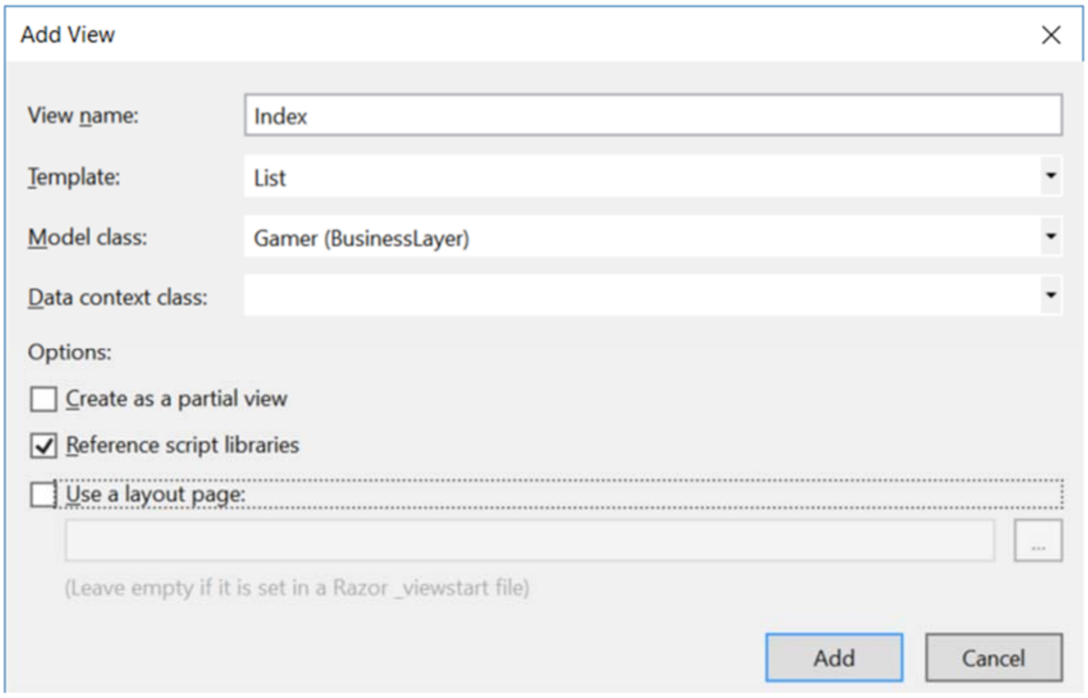

OnlineGame - Microsoft Visual Studio (Administrator)
File Edit View Project Build Incredibuild Debug Team tangibl
Debug Any CPU
Gamer.cs Gamer.cs GamerController.cs GamerBusiness
OnlineGame.Web OnlineGame.Web.
Test Explorer Server Explorer Toolbox
106 %
Go To View Ctrl+M, Ctrl+G
Add View...
Remove and Sort Usings Ctrl+R, Ctrl+G
Peek Definition
Go To Definition
Go To Implementation
Find All References
View Call Hierarchy Ctrl+K, Ctrl+T
Go To Declaration Ctrl+B
Navigate To... Ctrl+Shift+G
Navigate
Find Usages Alt+F7
Find Usages Advanced... Shift+Alt+F7
Refactor This... Ctrl+Shift+R
Refactor
Create Unit Tests
Inspect
Search with Pattern...
Cleanup Code Ctrl+Alt+F
Run To Cursor Ctrl+F10

```
            return View(gamer);
        }


        public ActionResult Index(int teamId
        {
            ////Entity Framework
            //OnlineGameContext context = ne
            //List<Gamer> gamers = context.G

            //Ado.Net
            GamerBusinessLayer gamerBusiness
            List<BusinessLayer.Gamer> gamers

            return View(gamers);
        }
    }
}
```


Add View

View name: Index

Template: List

Model class: Gamer (BusinessLayer)

Data context class:

Options:

- [ ] Create as a partial view
- [x] Reference script libraries
- [ ] Use a layout page:

(Leave empty if it is set in a Razor _viewstart file)

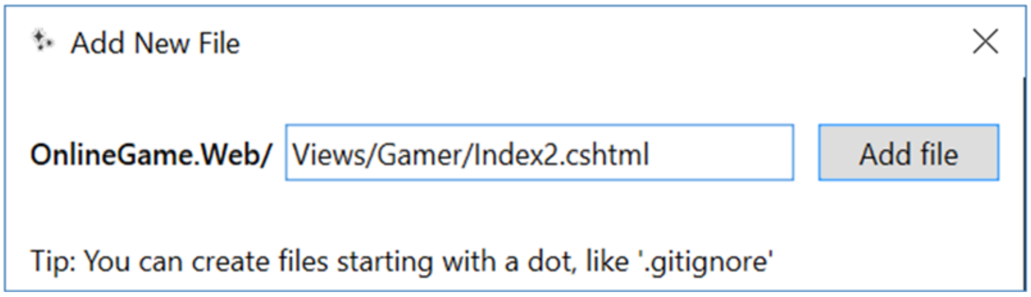
I personally don't like the previous way to create views,
because it will include some unnecessary changes in the source control.

Here is the way I usually do.

Because **Add New File** **(extension and update)**

press **Shift+F2**

Views/Gamer/Index2.cshtml


Add New File

OnlineGame.Web/ Views/Gamer/Index2.cshtml

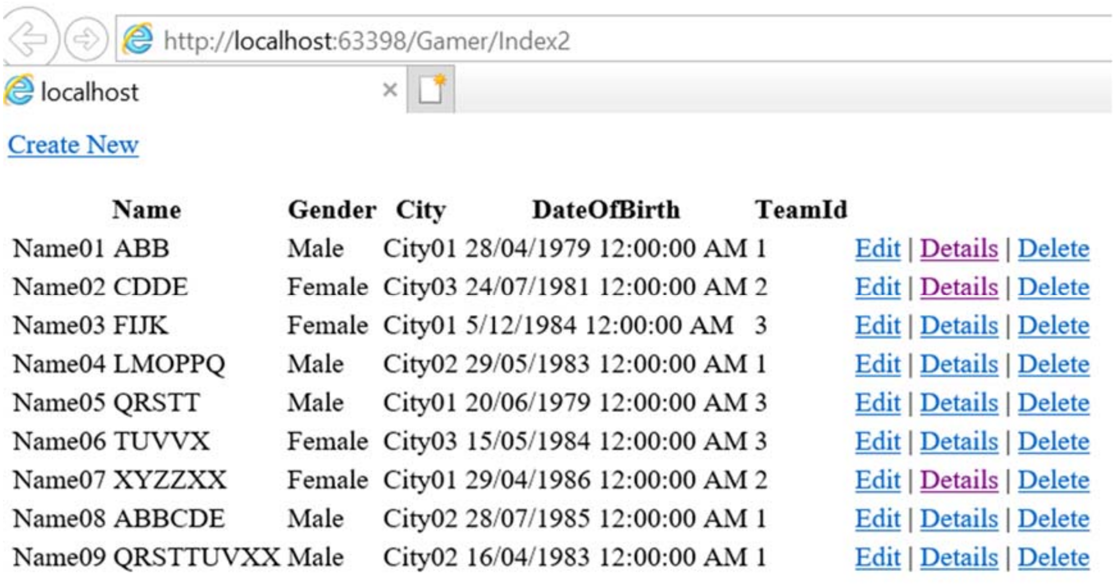
Tip: You can create files starting with a dot, like '.gitignore'

```
@model IEnumerable<BusinessLayer.Gamer>
@{
    Layout = null;
}
<p>
   @Html.ActionLink("Create New", "Create")
</p>
<table class="table">
   <tr>
       <th>
           @Html.DisplayNameFor(model => model.Name)
       </th>
       <th>
           @Html.DisplayNameFor(model => model.Gender)
       </th>
```

```
        <th>
            @Html.DisplayNameFor(model => model.City)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.DateOfBirth)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.TeamId)
        </th>
        <th></th>
    </tr>
    @foreach (var item in Model)
    {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.Name)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Gender)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.City)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.DateOfBirth)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.TeamId)
            </td>
            <td>
                @Html.ActionLink("Edit", "Edit", new { id = item.Id }) |
                @Html.ActionLink("Details", "Details", new { id = item.Id }) |
                @Html.ActionLink("Delete", "Delete", new { id = item.Id })
            </td>
        </tr>
    }
</table>
```

----

http://localhost:63398/Gamer/Index2

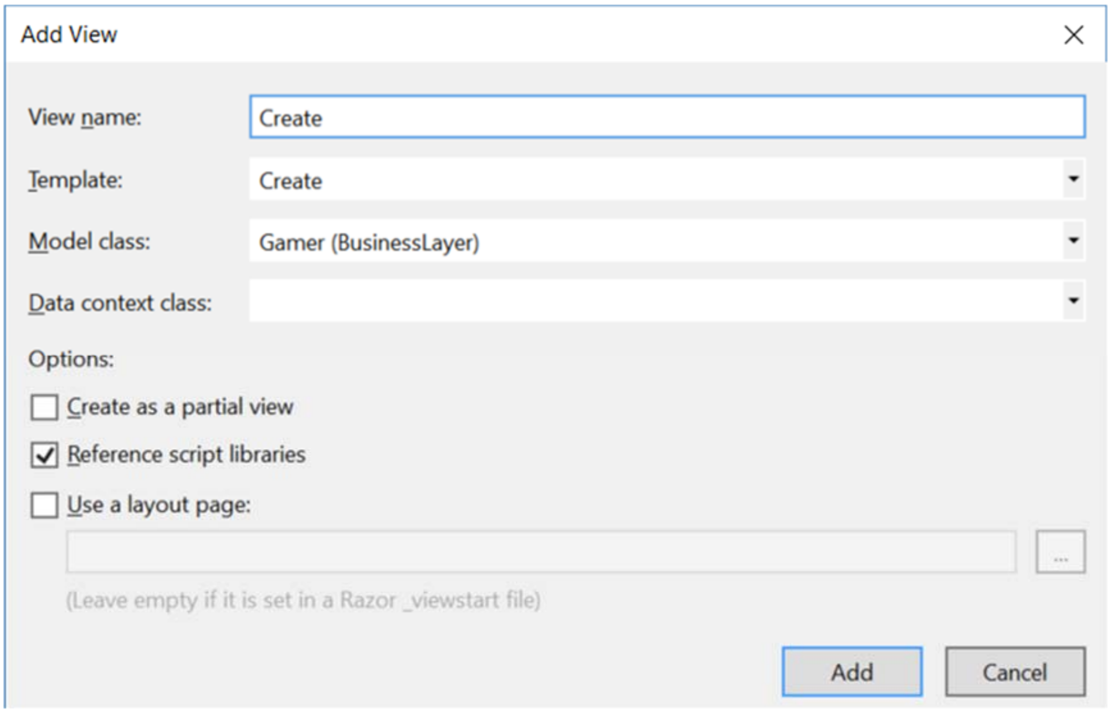

http://localhost:63398/Gamer/Index2

localhost

Create New

Name	Gender	City	DateOfBirth	TeamId	
Name01 ABB	Male	City01	28/04/1979 12:00:00 AM	1	Edit | Details | Delete
Name02 CDDE	Female	City03	24/07/1981 12:00:00 AM	2	Edit | Details | Delete
Name03 FIJK	Female	City01	5/12/1984 12:00:00 AM	3	Edit | Details | Delete
Name04 LMOPPQ	Male	City02	29/05/1983 12:00:00 AM	1	Edit | Details | Delete
Name05 QRSTT	Male	City01	20/06/1979 12:00:00 AM	3	Edit | Details | Delete
Name06 TUVVX	Female	City03	15/05/1984 12:00:00 AM	3	Edit | Details | Delete
Name07 XYZZXX	Female	City01	29/04/1986 12:00:00 AM	2	Edit | Details | Delete
Name08 ABBCDE	Male	City02	28/07/1985 12:00:00 AM	1	Edit | Details | Delete
Name09 QRSTTUVXX	Male	City02	16/04/1983 12:00:00 AM	1	Edit | Details | Delete

4.4. Create: OnlineGame.Web/Views/Gamer/Create.cshtml


Add View

View name: Create

Template: Create

Model class: Gamer (BusinessLayer)

Data context class:

Options:

[ ] Create as a partial view

[✓] Reference script libraries

[ ] Use a layout page:

(Leave empty if it is set in a Razor _viewstart file)

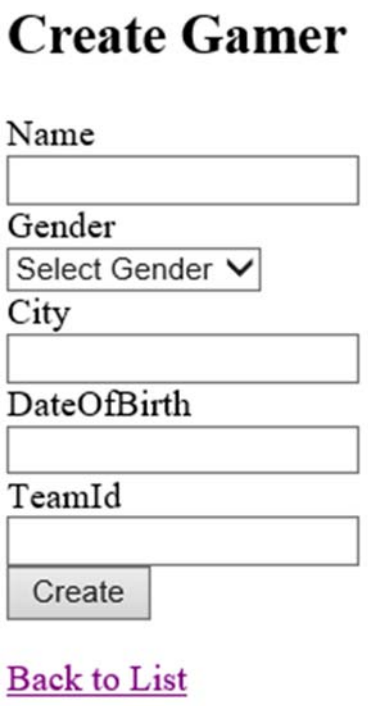
```
@model BusinessLayer.Gamer
@{
}
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script src="~/Scripts/jquery.validate.min.js"></script>
<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>
<div class="row">
    <h2>Create Gamer</h2>
    @*@using (Html.BeginForm())*@
    @*@using (Html.BeginForm("Create", "Gamer"))*@
    @*@using (Html.BeginForm("Create2", "Gamer"))*@
     @*@using (Html.BeginForm("Create4", "Gamer"))*@
    @using (Html.BeginForm("Create3", "Gamer"))
    {
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-horizontal">
            <div class="form-group">
                @Html.LabelFor(model => model.Name, new { @class = "control-label col-md-2" })
                <div class="col-md-10">
                    @Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-
control" } })
                    @Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })
                </div>
            </div>
            <div class="form-group">
                @Html.LabelFor(model => model.Gender, new { @class = "control-label col-md-2" })
                <div class="col-md-10">
                    @*@Html.EditorFor(model => model.Gender, new { htmlAttributes = new { @class = "form-
control" } })*@
                    @*@Html.DropDownList("Gender",new List<SelectListItem>
                        {
                            new SelectListItem{Text = "Male", Value = "Male"},
                            new SelectListItem{Text = "Female", Value = "Female"}
                        })*@
                    @Html.DropDownList("Gender", new List<SelectListItem>
                    {
                        new SelectListItem{Text = "Male", Value = "Male"},
                        new SelectListItem{Text = "Female", Value = "Female"}
                    }, "Select Gender")
                    @Html.ValidationMessageFor(model => model.Gender, "", new { @class = "text-danger" })
                </div>
            </div>
            <div class="form-group">
                @Html.LabelFor(model => model.City, new { @class = "control-label col-md-2" })
                <div class="col-md-10">
                    @Html.EditorFor(model => model.City, new { htmlAttributes = new { @class = "form-
control" } })
                    @Html.ValidationMessageFor(model => model.City, "", new { @class = "text-danger" })
                </div>
            </div>
            <div class="form-group">
                @Html.LabelFor(model => model.DateOfBirth, new { @class = "control-label col-md-2" })
                <div class="col-md-10">
                    @Html.EditorFor(model => model.DateOfBirth, new { htmlAttributes = new { @class = "form-
control" } })
                    @Html.ValidationMessageFor(model => model.DateOfBirth, "", new { @class = "text-
danger" })
                </div>
            </div>
            <div class="form-group">
                @Html.LabelFor(model => model.TeamId, new { @class = "control-label col-md-2" })
                <div class="col-md-10">
                    @Html.EditorFor(model => model.TeamId, new { htmlAttributes = new { @class = "form-
control" } })
```

```
                    @*@Html.ValidationMessageFor(model => model.TeamId, "", new { @class = "text-
danger" })*@
                </div>
            </div>
            <div class="form-group">
                <div class="col-md-offset-2 col-md-10">
                    <input type="submit" value="Create" class="btn btn-default" />
                </div>
            </div>
        </div>
    }
    <div>
        @Html.ActionLink("Back to List", "Index2")
    </div>
</div>
@*
1.
1.1.
Select <form>....</form>
We are using Web Essentials extension
Select part of HTML text, then press Shift + Alt + W
it will surround those pieces of text with <div>
1.2.
// <div class="row">
class="row" in bootstrap means this is the container for the bootstrap 12 columns grid system
1.3.
// <div class="col-md-6">
1.3.1.
.col-lg-XX means the screen size is >= 1200px, lg means large
.col-md-XX means the screen size is >= 992px, means 992px to 1199px, md means medium
.col-sm-XX means the screen size is >= 768px, means 768px to 991px, sm means small
.col-xs-XX means the screen size is < 768px, xs means extra small
1.3.2.
// <div class="col-md-6">
The md in   class="col-md-6"  means the screen size is 992px to 1199px
12 columns grid system divide this screen to 12 columns,
The 6 in   class="col-md-6" means this div occupy 6 columns out of 12 columns grid system.
Therefore,
// <div class="col-md-6">
//     ...
// </div>
// <div class="col-md-6">
//     <h2>The Map</h2>
// </div>
It will become left half and right half when it is full screen.
However, when reduce screen size less than 11 columns,
then it will become top and bottom.
1.3.3.
If we do this
// <div class="col-md-6 col-xs-8">
//     ...
// </div>
// <div class="col-md-6 col-xs-4">
//     <h2>The Map</h2>
// </div>
The  .col-xs-XX   will override  .col-md-XX
However, we actually only need big one, .col-md-XX
because for small screen device like phone,
we want it automatically align vertically.
so let's only use   .col-md-XX   and delete   .col-xs-XX
2.
// <div class="form-group">
//    <label>Date</label>
//    <input class="form-control"/>
// </div>
// ...
// <input type="submit" value="Add" class="btn btn-success"/>
2.1.
.form-group in bootstrap normally used for group label and input.
2.2.
.form-control in bootstrap will occupy the 100% width
2.3.
.btn-success in bootstrap will become green btn
2.4.
btn-danger in bootstrap will become red btn
2.5.
btn-warning in bootstrap will become orange btn
2.6.
btn-default in bootstrap will become gray btn


3.
3.1.
//<input type="submit" value="Create" class="btn btn-default" />
It is the submit button of the form.
3.2.
//@using (Html.BeginForm())
//@using (Html.BeginForm("Create", "Gamer"))
These two using will create
//<form action="/Gamer/Create3" method="post" novalidate="novalidate">
It will run the "Gamer" controller and "Create" HttpPost action.
//@using (Html.BeginForm())
It does not specify any controller or any action.
Thus, it will use the default controller and default HttpPost action.
It is Create.cshtml in Views/Gamer folder.
Thus, it will run "Gamer" controller and "Create" HttpPost action
3.3.
//@using (Html.BeginForm("Create2", "Gamer"))
It will create
//<form action="/Gamer/Create2" method="post" novalidate="novalidate">
It will run "Gamer" controller and "Create2" HttpPost action.
4.
4.1.
//@Html.LabelFor(model => model.Name, new { @class = "control-label col-md-2" })
It will create
//<label class="control-label col-md-2" for="Name">Name</label>
4.2.
//@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })
It will create
//<input class="form-control text-box single-line" id="Name" name="Name" type="text" value="">
4.3.
//@Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })
It will create
//<span class="field-validation-valid text-danger" data-valmsg-for="Name" data-valmsg-
replace="true"></span>
The cshtml also use the following JS
//<script src="~/Scripts/jquery-1.10.2.min.js"></script>
//<script src="~/Scripts/jquery.validate.min.js"></script>
//<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>
Thus, the span will display the error message of Name input.
4.4.
//@Html.DropDownList("Gender",new List<SelectListItem>
//{
//    new SelectListItem{Text = "Male", Value = "Male"},
//    new SelectListItem{Text = "Female", Value = "Female"}
//})
It will create
//<select id="Gender" name="Gender">
//    <option value="Male">Male</option>
//    <option value="Female">Female</option>
//</select>
```

```
4.5.
//@Html.DropDownList("Gender", new List<SelectListItem>
//{
//    new SelectListItem{Text = "Male", Value = "Male"},
//    new SelectListItem{Text = "Female", Value = "Female"}
//}, "Select Gender")
It will create
//<select id="Gender" name="Gender">
//    <option value="">Select Gender</option>
//    <option value="Male">Male</option>
//    <option value="Female">Female</option>
//</select>
4.6.
//@Html.ActionLink("Back to List", "Index2")
It will create
//<a href="/Gamer/Index2">Back to List</a>
*@
```

http://localhost:63398/Gamer/Create


Create Gamer

Name

Gender

Select Gender

City

DateOfBirth

TeamId