(T8)討論 RoutePrefixAttribute、RouteAttribute、RouteName、RouteConstraints。比較 IHttpActionResult、HttpResponseMessage
CourseGUID 4c5822ff-7111-4e25-a336-ef18d48d54bd
==

==

0. What to Learn

The tutorial will discuss ...
How to use RoutePrefixAttribute, RouteAttribute, RouteName, RouteConstraints.
IHttpActionResult V.S. HttpResponseMessage

本堂課討論
關於 RoutePrefixAttribute、RouteAttribute、RouteName、RouteConstraints。
比較 IHttpActionResult 和 HttpResponseMessage

6.
Attribute routing

6.1.
E.g.
//public async Task<IHttpActionResult> GetGamer(int id){...}
....
//[Route("api/gamer/{id}/skills")]
//public async Task<IHttpActionResult> GetGamerSkills(int id){...}
When we call "api/gamer/1" and if we don't have Route attribute,
the API will be confused,
because both GetGamerSkills() and GetGamer() can map to "api/gamer/1".
Thus, we need Route attribute
[Route("api/gamer/{id}/skills")] will make GetGamerSkills() map to something like "api/gamer/1/skills".
Thus, GetGamer() can map to something like "api/gamer/1".

6.2.
In this case,
GetGamer() is using Convention-based routing.
GetGamerSkills() is using Attribute Routing.

6.3.
In
OnlineGame.WebApi/WebApiConfig.cs/WebApiConfig.cs
//config.MapHttpAttributeRoutes();
It enables Attribute Routing.

7.
RoutePrefix and Route attribute
//[RoutePrefix("api/gamer2")]
RoutePrefix attribute is for route prefix at the controller level.
Route attribute use that route prefix plus its own route value.
//[Route("~/api/getGamerSkillsByGamerId/{gamerId}")]
if you want to override the route prefix,
just use ~ (tilde) symbol

8.
attribute routing constraints
Reference:
https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/attribute-routing-in-web-api-2#route-constraints

Routing constraints can apply to decimal, double, float, long, bool...etc.

8.1.
```
//// GET: api/gamer3/GetGamerBySomething/2
//[Route("GetGamerBySomething/{gamerId:int}")]
//public async Task<IHttpActionResult> GetGamerBySomething(int gamerId)
```
int means integer
...
```
//// GET: api/gamer3/GetGamerBySomething/male
////[Route("GetGamerBySomething/{gender:string}")]    //Error, string type is  not valid
//[Route("GetGamerBySomething/{gender:alpha}")]
//public async Task<IHttpActionResult> GetGamerBySomething(string gender)
```
alpha means uppercase or lowercase alphabet.

8.2.
```
//[Route("getGamerById/{gamerId:int:min(2)}")]
//public async Task<IHttpActionResult> GetGamerById(int gamerId)
GET: api/gamer3/getGamerById/1
```
gamerId must be int and min is 2

8.3.
```
//[Route("getGamerById2/{gamerId:int:min(2):max(5)}")]
//public async Task<IHttpActionResult> GetGamerById2(int gamerId)
GET: api/gamer3/getGamerById2/1
```
gamerId must be int and min is 2, max is 5

8.4.
```
//[Route("getGamerById3/{gamerId:range(2,5)}")]
//public async Task<IHttpActionResult> GetGamerById3(int gamerId)
GET: api/gamer3/getGamerById3/1
```
gamerId must be int and min is 2, max is 5

8.5.
```
////[Route("getGamersByGender/{gender:string}")]    //Error, string type is not  valid
//[Route("getGamersByGender/{gender:alpha}")]
//public async Task<IHttpActionResult> GetGamersByGender(string gender)
```
alpha means uppercase or lowercase alphabet characters.
```
GET: api/gamer3/getGamersByGender/female
```

8.7.
```
//[Route("getGamersByGender3/{gender:alpha:maxlength(5)}")]
//public async Task<IHttpActionResult> GetGamersByGender3(string gender)
GET: api/gamer3/getGamersByGender3/female       //404
GET: api/gamer3/getGamersByGender3/male
```
alpha means uppercase or lowercase alphabet characters.
max alpha length is 5

8.8.
```
//[Route("getGamersByGender3/{gender:alpha:maxlength(5)}")]
//public async Task<IHttpActionResult> GetGamersByGender3(string gender)
GET: api/gamer3/getGamersByGender3/female       //404
GET: api/gamer3/getGamersByGender3/male
```
alpha means uppercase or lowercase alphabet characters.

max alpha length is 5

8.9.
//[Route("getGamersByGender4/{gender:alpha:minlength(5):maxlength(7)}")]
//public async Task<IHttpActionResult> GetGamersByGender4(string gender)
GET: api/gamer3/getGamersByGender4/female
GET: api/gamer3/getGamersByGender4/male   //404
alpha means uppercase or lowercase alphabet characters.
max alpha length is 7, and min length is 5.

9.
Route names
9.1.
E.g.
//[Route("{id:int}", Name = "GetGamerById")]
//public async Task<IHttpActionResult> GetGamerById(int id)
...
//HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.Created);
//response.Headers.Location = new
//   Uri(Url.Link("GetGamerById", new { id = gamer.Id }));
...
//return CreatedAtRoute("GetGamerById", new { id = gamer.Id }, gamer);   //Created/201
9.2.
//return CreatedAtRoute("DefaultApi", new { id = gamer.Id }, gamer);   //Created/201
...
//HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.Created);
//response.Headers.Location = new Uri(Request.RequestUri + "/" + gamer.Id);

10.
IHttpActionResult vs HttpResponseMessage
10.1.
IHttpActionResult
10.1.1.
HttpResponseMessage is from Web API 1
IHttpActionResult is from Web API 2
10.1.2.
IHttpActionResult make code cleaner.
10.1.3.
The following type implements IHttpActionResult interface.
Unauthorized()
BadRequest()
NotFound()
Created()
OK()
InternalServerError()

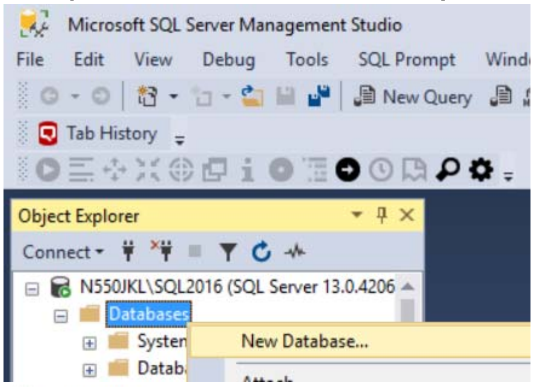
1. OnlineGame2 DB

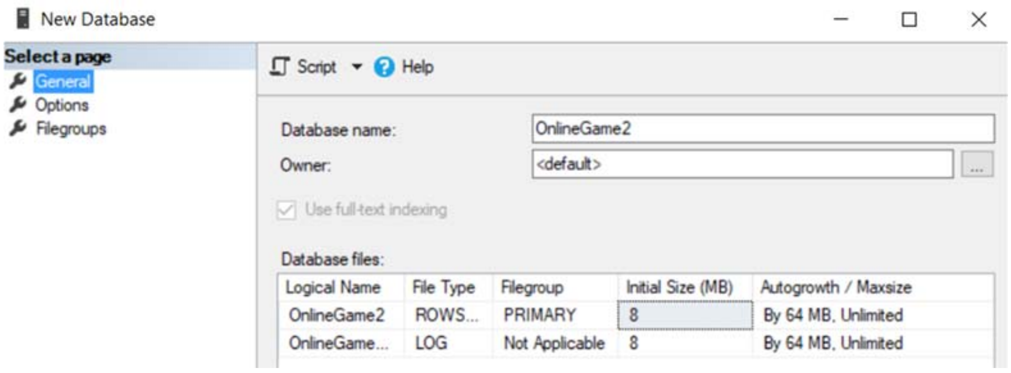
1.0. Some points

1.
Regular expression
https://regexr.com/
2.
Calling Stored Procedure from Entity Framework 6 Code First
http://www.dotnetodyssey.com/2015/03/12/calling-stored-procedure-from-entity-framework-6-code-first/

1.1. TSQL

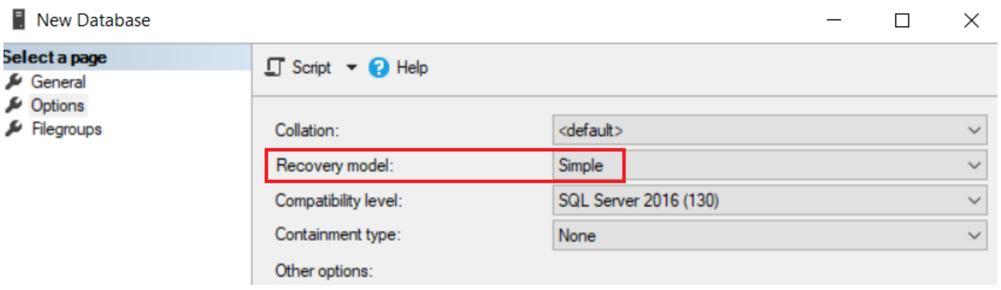
In SQL server Management Studio (SSMS)
Database --> Right Click --> New Database -->
In General Tab -->
Name: **OnlineGame2**
In options Tab --> Recovery model : **Simple**





Database name: OnlineGame2
Owner: <default>
Use full-text indexing
Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize
OnlineGame2	ROWS...	PRIMARY	8	By 64 MB, Unlimited
OnlineGame...	LOG	Not Applicable	8	By 64 MB, Unlimited



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

```
--1.1 ------------------------------------------------------------
--Drop Table if it exists.
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'GamerSkill' ) )
    BEGIN
        TRUNCATE TABLE GamerSkill;
        DROP TABLE GamerSkill;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Skill' ) )
    BEGIN
        TRUNCATE TABLE Skill;
        DROP TABLE Skill;
    END;
GO -- Run the previous command and begins new batch
--IF OBJECT_ID('Gamer') IS NOT NULL
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 OBJECT_ID('Gamer') IS NOT NULL
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

--1.2 ----------------------------------------------------------
--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 = 'spInsertGamerSkill' ) )
    BEGIN
        DROP PROCEDURE spInsertGamerSkill;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.ROUTINES
              WHERE     ROUTINE_TYPE = 'PROCEDURE'
                        AND LEFT(ROUTINE_NAME, 3) NOT IN ( 'sp_', 'xp_', 'ms_' )
                        AND SPECIFIC_NAME = 'spDeleteGamerSkill' ) )
    BEGIN
        DROP PROCEDURE spDeleteGamerSkill;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.ROUTINES
              WHERE     ROUTINE_TYPE = 'PROCEDURE'
                        AND LEFT(ROUTINE_NAME, 3) NOT IN ( 'sp_', 'xp_', 'ms_' )
                        AND SPECIFIC_NAME = 'spSelectGamerSkill' ) )
    BEGIN
        DROP PROCEDURE spSelectGamerSkill;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
                FROM      INFORMATION_SCHEMA.ROUTINES
                WHERE     ROUTINE_TYPE = 'PROCEDURE'
                          AND LEFT(ROUTINE_NAME, 3) NOT IN ( 'sp_', 'xp_', 'ms_' )
                          AND SPECIFIC_NAME = 'spSkillsAssignToTheGamer' ) )
    BEGIN
        DROP PROCEDURE spSkillsAssignToTheGamer;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
                FROM      INFORMATION_SCHEMA.ROUTINES
                WHERE     ROUTINE_TYPE = 'PROCEDURE'
                          AND LEFT(ROUTINE_NAME, 3) NOT IN ( 'sp_', 'xp_', 'ms_' )
                          AND SPECIFIC_NAME = 'spSkillsNotAssignToTheGamer' ) )
    BEGIN
        DROP PROCEDURE spSkillsNotAssignToTheGamer;
    END;
GO -- Run the previous command and begins new batch

--2 ----------------------------------------------------------
CREATE TABLE Team
    (
      Id INT PRIMARY KEY
             IDENTITY(1, 1)
             NOT NULL ,
     Name NVARCHAR(50) NOT NULL
    );
GO -- Run the previous command and begins new batch
CREATE TABLE Gamer
    (
      Id INT PRIMARY KEY
             IDENTITY(1, 1)
             NOT NULL ,
     Name NVARCHAR(50) NOT NULL ,
      Gender NVARCHAR(50) NOT NULL ,
      Score INT NOT NULL ,
      TeamId INT FOREIGN KEY REFERENCES Team ( Id )
    );
GO -- Run the previous command and begins new batch
CREATE TABLE Skill
    (
      Id INT PRIMARY KEY
             IDENTITY(1, 1)
             NOT NULL ,
     Name NVARCHAR(50) NOT NULL
    );
GO -- Run the previous command and begins new batch
CREATE TABLE GamerSkill
```

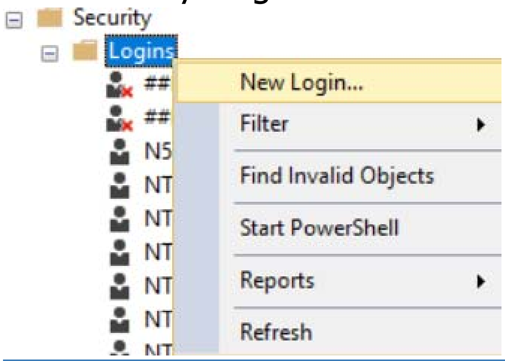
```
    (
       GamerId INT FOREIGN KEY REFERENCES Gamer ( Id )
                    NOT NULL ,
       SkillId INT FOREIGN KEY REFERENCES Skill ( Id )
                    NOT NULL ,
       CreatedDate DATETIME DEFAULT ( GETUTCDATE() )
                              NOT NULL ,
      PRIMARY KEY ( GamerId, SkillId )
    );
GO -- Run the previous command and begins new batch

--3 ---------------------------------------------------------
INSERT  Team
VALUES  ( 'TeamOne' );
INSERT  Team
VALUES  ( 'TeamTwo' );
INSERT  Team
VALUES  ( 'TeamThree' );
GO -- Run the previous command and begins new batch

INSERT  INTO Gamer
VALUES  ( 'NameOne ABC', 'Male', 5000, 1 );
INSERT  INTO Gamer
VALUES  ( 'NameTwo ABCDE', 'Female', 4500, 1 );
INSERT  INTO Gamer
VALUES  ( 'NameThree EFGH', 'Male', 6500, 3 );
INSERT  INTO Gamer
VALUES  ( 'NameFour HIJKLMN', 'Female', 45000, 2 );
INSERT  INTO Gamer
VALUES  ( 'NameFive NOP', 'Male', 3000, 3 );
INSERT  INTO Gamer
VALUES  ( 'NameSix PQRSTUVW', 'Male', 4000, 3 );
INSERT  INTO Gamer
VALUES  ( 'NameSeven XYZ', 'Male', 4500, 1 );
GO -- Run the previous command and begins new batch

INSERT  INTO Skill
VALUES  ( 'SkillA Play Dead' );
INSERT  INTO Skill
VALUES  ( 'SkillB Flame Punch' );
INSERT  INTO Skill
VALUES  ( 'SkillC Steal' );
INSERT  INTO Skill
VALUES  ( 'SkillD Fly' );
INSERT  INTO Skill
VALUES  ( 'SkillE Super Speed' );
INSERT  INTO Skill
VALUES  ( 'SkillF Forzen' );
INSERT  INTO Skill
VALUES  ( 'SkillG Invisible' );
GO -- Run the previous command and begins new batch

INSERT  INTO GamerSkill
        ( GamerId, SkillId )
```

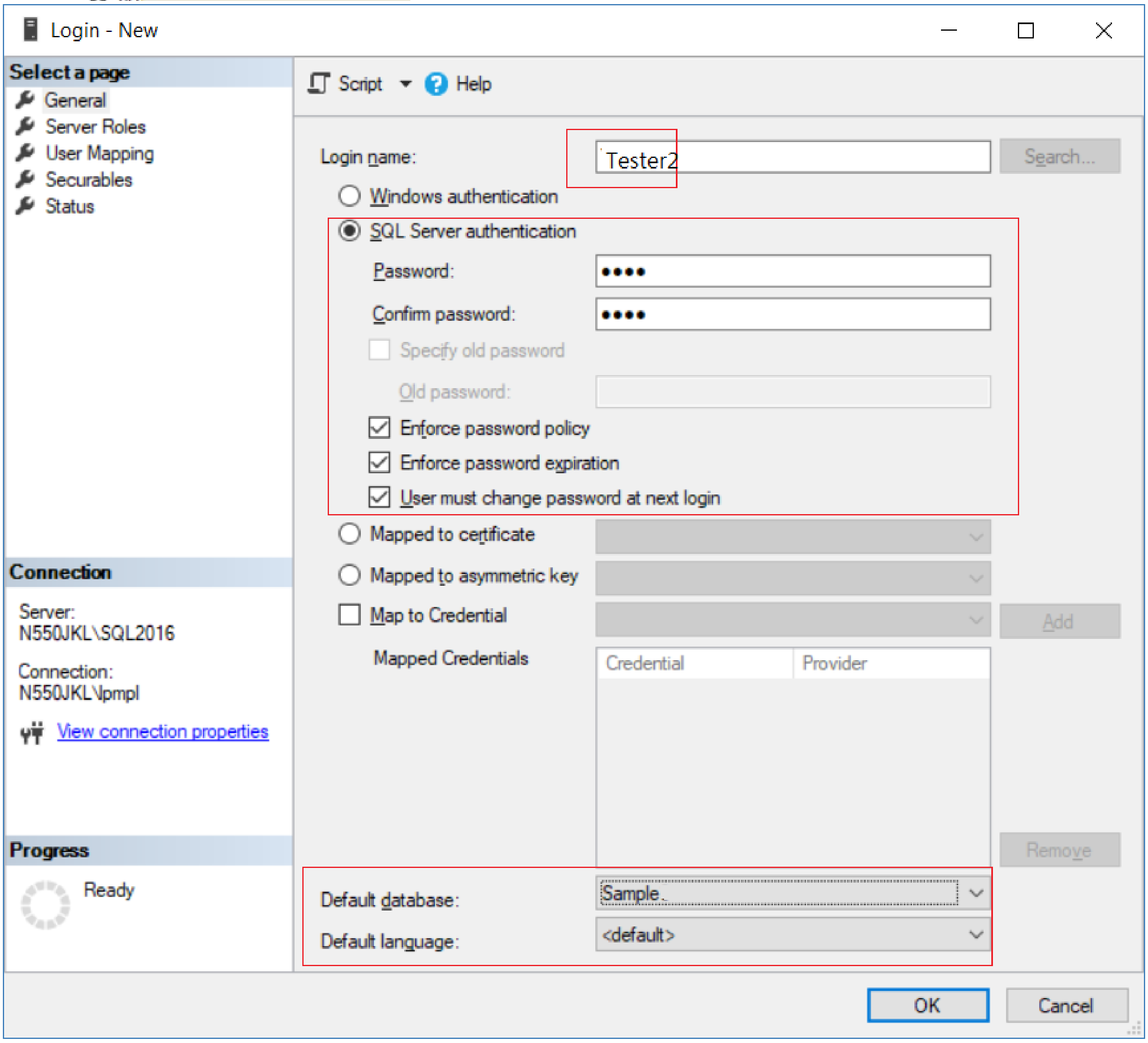
```sql
VALUES  ( 1, 2 );
INSERT  INTO GamerSkill
        ( GamerId, SkillId )
VALUES  ( 1, 3 );
INSERT  INTO GamerSkill
        ( GamerId, SkillId )
VALUES  ( 2, 2 );
INSERT  INTO GamerSkill
        ( GamerId, SkillId )
VALUES  ( 2, 1 );
INSERT  INTO GamerSkill
        ( GamerId, SkillId )
VALUES  ( 2, 4 );
GO -- Run the previous command and begins new batch

--4 SP ----------------------------------------------------------
CREATE PROCEDURE spInsertGamerSkill
    (
      @GamerId INT ,
      @SkillId INT
    )
AS
    BEGIN
        INSERT  INTO GamerSkill
                ( GamerId, SkillId )
        VALUES  ( @GamerId, -- GamerId - int
                  @SkillId  -- SkillId - int
                  );
    END;
GO -- Run the previous command and begins new batch
CREATE PROCEDURE spDeleteGamerSkill
    (
      @GamerId INT ,
      @SkillId INT
    )
AS
    BEGIN
        DELETE  FROM GamerSkill
        WHERE   GamerId = @GamerId
                AND SkillId = @SkillId;
    END;
GO -- Run the previous command and begins new batch
CREATE PROCEDURE spSelectGamerSkill
AS
    BEGIN
        SELECT  gs.GamerId ,
                g.Name ,
                g.Gender ,
                g.Score ,
                gs.SkillId ,
                s.Name
        FROM    Gamer g
                INNER JOIN GamerSkill gs ON g.Id = gs.GamerId
                INNER JOIN Skill s ON s.Id = gs.SkillId;
    END;
GO -- Run the previous command and begins new batch
--This is for test purpose
--If you want to use it in EF, you have to return a view or table function.
CREATE PROCEDURE spSkillsAssignToTheGamer ( @GamerId INT )
AS
    BEGIN
        SELECT  *
        FROM    GamerSkill gs
                INNER JOIN Skill s ON s.Id = gs.SkillId
        WHERE   GamerId = @GamerId;
    END;
GO -- Run the previous command and begins new batch
--This is for test purpose
--If you want to use it in EF, you have to return a view or table function.
CREATE PROCEDURE spSkillsNotAssignToTheGamer ( @GamerId INT )
AS
    BEGIN
        SELECT  *
        FROM    Skill s
        WHERE   s.Id NOT IN (
                SELECT  s.Id
                FROM    GamerSkill gs
                        INNER JOIN Skill s ON s.Id = gs.SkillId
                WHERE   GamerId = @GamerId );
    END;
GO -- Run the previous command and begins new batch
--This is for test purpose
--If you want to use it in EF, you have to return a view or table function.
--EXEC spInsertGamerSkill @GamerId = 100, @SkillId = 1;
--EXEC spDeleteGamerSkill @GamerId = 100, @SkillId = 1;
--EXEC spSelectGamerSkill
--EXEC spSkillsAssignToTheGamer @GamerId=1
--EXEC spSkillsNotAssignToTheGamer @GamerId=1
```

1.2. Security login

In SQL server
Object Explorer --> Security --> Logins --> New Logins
-->
General Tab
Login Name :
Tester2
Password:
1234
Default Database:
OnlineGame
-->
Server Roles Tab
Select
sysadmin

-->
User Mapping Tab
Select **OnlineGame**
Select every single role.





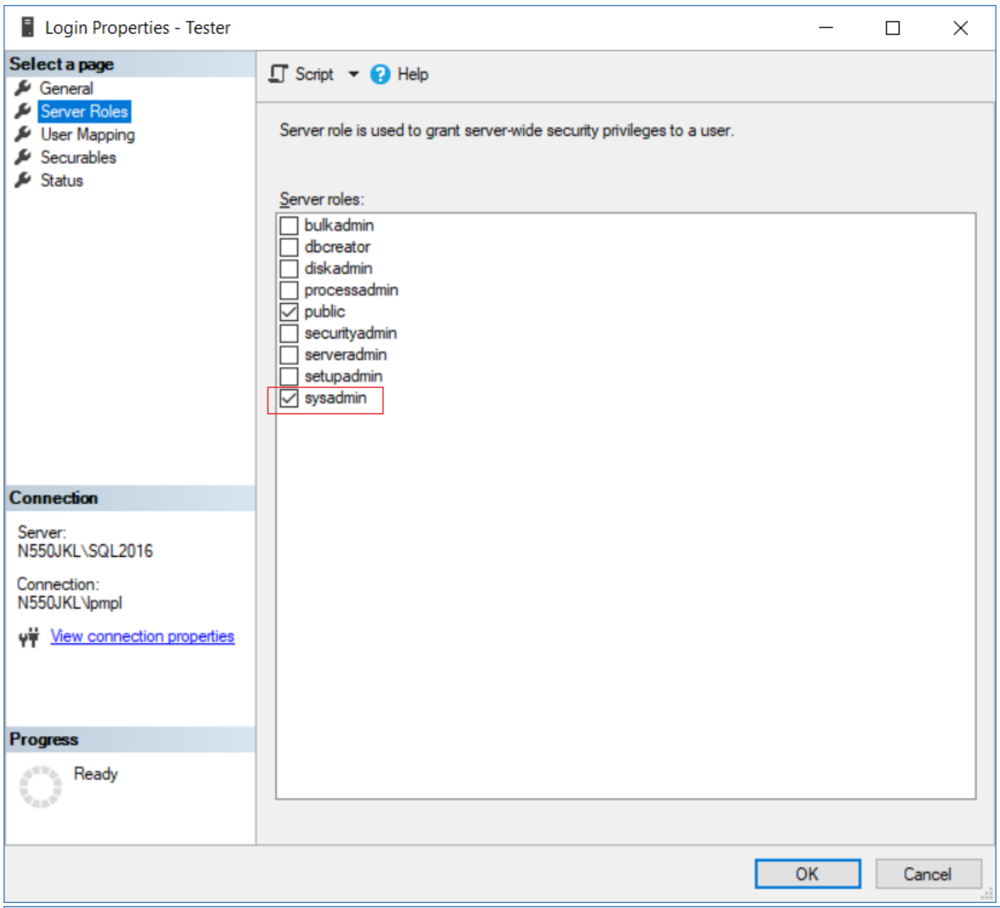
Login name: Tester2
Windows authentication
SQL Server authentication
Password: ••••
Confirm password: ••••
Specify old password
Old password:
Enforce password policy ✓
Enforce password expiration ✓
User must change password at next login ✓
Mapped to certificate
Mapped to asymmetric key
Map to Credential
Mapped Credentials

Credential	Provider

Default database: Sample
Default language: <default>

Connection
Server:
N550JKL\SQL2016
Connection:
N550JKL\lpmpl
View connection properties

Progress
Ready



Login Properties - Tester

Select a page
- General
- Server Roles
- User Mapping
- Securables
- Status

Server role is used to grant server-wide security privileges to a user.

Server roles:

- [ ] bulkadmin
- [ ] dbcreator
- [ ] diskadmin
- [ ] processadmin
- [x] public
- [ ] securityadmin
- [ ] serveradmin
- [ ] setupadmin
- [x] sysadmin

Connection

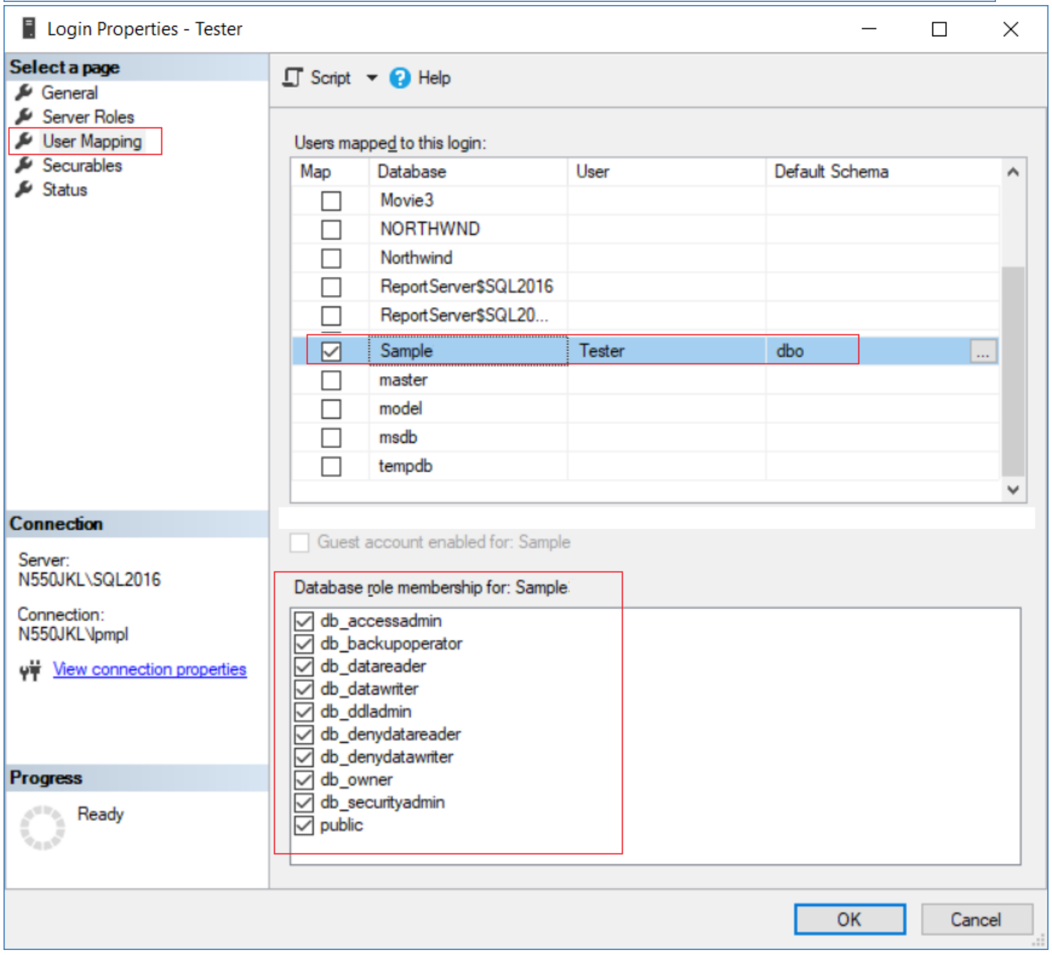
Server:
N550JKL\SQL2016

Connection:
N550JKL\lpmpl

View connection properties

Progress

Ready



Login Properties - Tester

Select a page
- General
- Server Roles
- User Mapping
- Securables
- Status

Users mapped to this login:

Map	Database	User	Default Schema
☐	Movie3		
☐	NORTHWND		
☐	Northwind		
☐	ReportServer$SQL2016		
☐	ReportServer$SQL20...		
✓	Sample	Tester	dbo
☐	master		
☐	model		
☐	msdb		
☐	tempdb		

- [ ] Guest account enabled for: Sample

Database role membership for: Sample

- [x] db_accessadmin
- [x] db_backupoperator
- [x] db_datareader
- [x] db_datawriter
- [x] db_ddladmin
- [x] db_denydatareader
- [x] db_denydatawriter
- [x] db_owner
- [x] db_securityadmin
- [x] public

Connection

Server:
N550JKL\SQL2016

Connection:
N550JKL\lpmpl

View connection properties

Progress

Ready

2. OnlineGame Solution

2.1. OnlineGame Solution

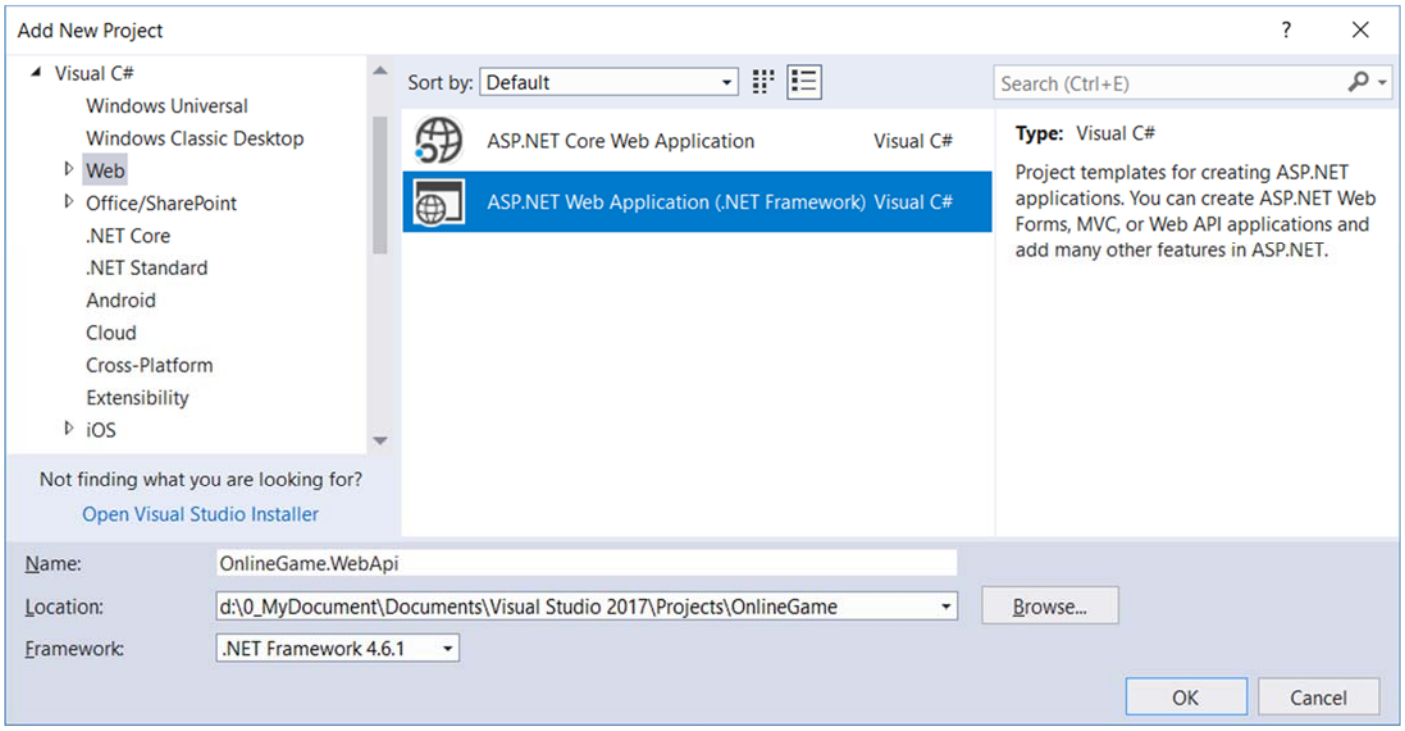
File --> New --> Project... -->

Other Project Types --> Visual Studio Solutions --> Blank Solution
-->
Name: **OnlineGame**

2.2. OnlineGame.WebApi

Solutions Name --> Add --> New Project -->

Visual C# --> Web --> ASP.NET Web Application (.Net Framework)
-->
Name: **OnlineGame.WebApi**
--> Select "**Web API**"
-->
Change Authentication
-->
Individual User Accounts
-->
OK


Add New Project

Visual C#: Windows Universal, Windows Classic Desktop, Web, Office/SharePoint, .NET Core, .NET Standard, Android, Cloud, Cross-Platform, Extensibility, iOS

Not finding what you are looking for?
Open Visual Studio Installer

Sort by: Default

ASP.NET Core Web Application — Visual C#
ASP.NET Web Application (.NET Framework) — Visual C#

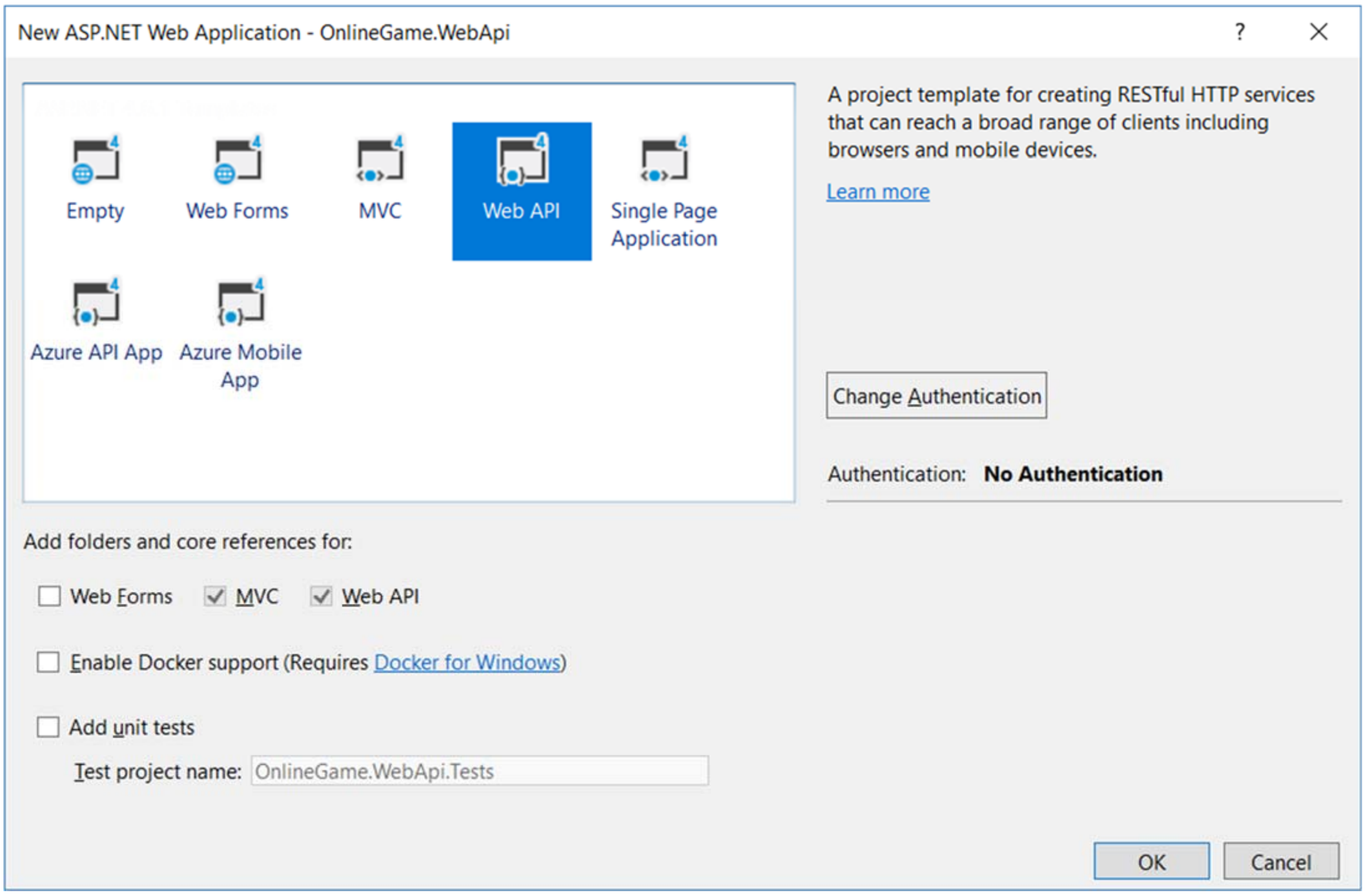
Search (Ctrl+E)

Type: Visual C#

Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

Name: OnlineGame.WebApi
Location: d:\0_MyDocument\Documents\Visual Studio 2017\Projects\OnlineGame
Framework: .NET Framework 4.6.1

Browse...
OK
Cancel


New ASP.NET Web Application - OnlineGame.WebApi

A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.

Learn more

Authentication: **No Authentication**

Add folders and core references for:

- [ ] Web Forms
- [x] MVC
- [x] Web API

- [ ] Enable Docker support (Requires Docker for Windows)

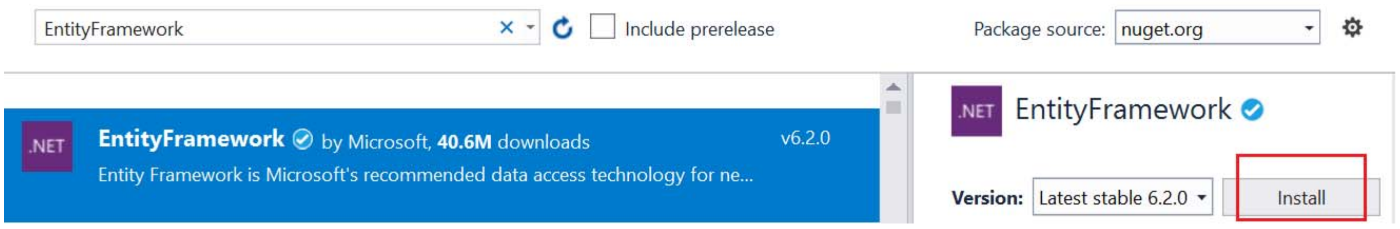
- [ ] Add unit tests

Test project name: OnlineGame.WebApi.Tests

3. OnlineGame.WebApi - Entity Framework

3.1. Install Entity Framework

Tools --> NuGet Package Manager --> Manage NuGet Packages for Solutions...
--> Browse tab --> Search : **EntityFramework**
--> Install it


EntityFramework

- [ ] Include prerelease

Package source: nuget.org

EntityFramework by Microsoft, **40.6M** downloads v6.2.0
Entity Framework is Microsoft's recommended data access technology for ne...

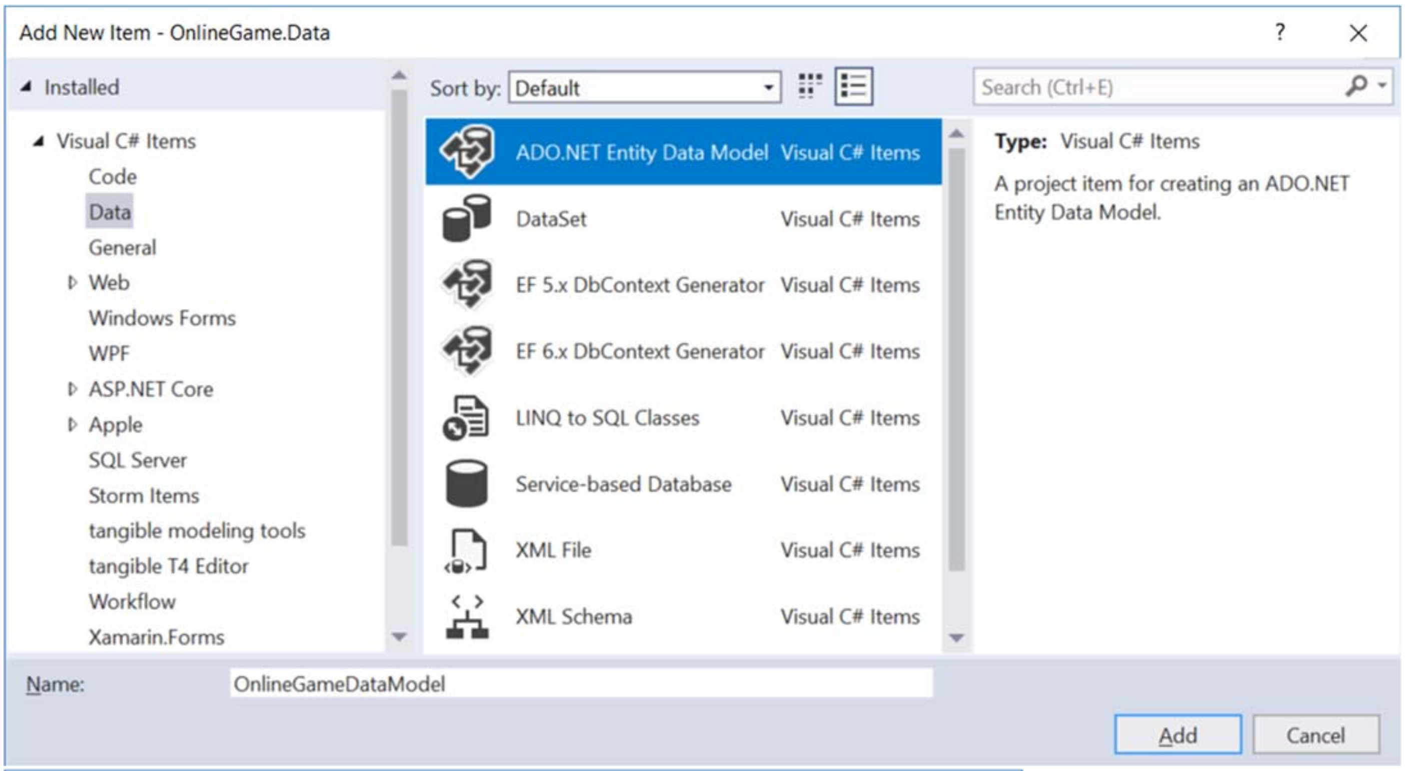
EntityFramework

Version: Latest stable 6.2.0

3.2. ADO.Net Entity Data Model - Entity Framework

In Visual Studio 2017
Models Folder --> Right Click --> Add --> New Item
--> Visual C# --> Data --> ADO.Net Entity Data Model
Name:
OnlineGameDataModel
-->
EF Designer from database

....
-->
Save Connection settings in Web.Config as:
OnlineGameContext

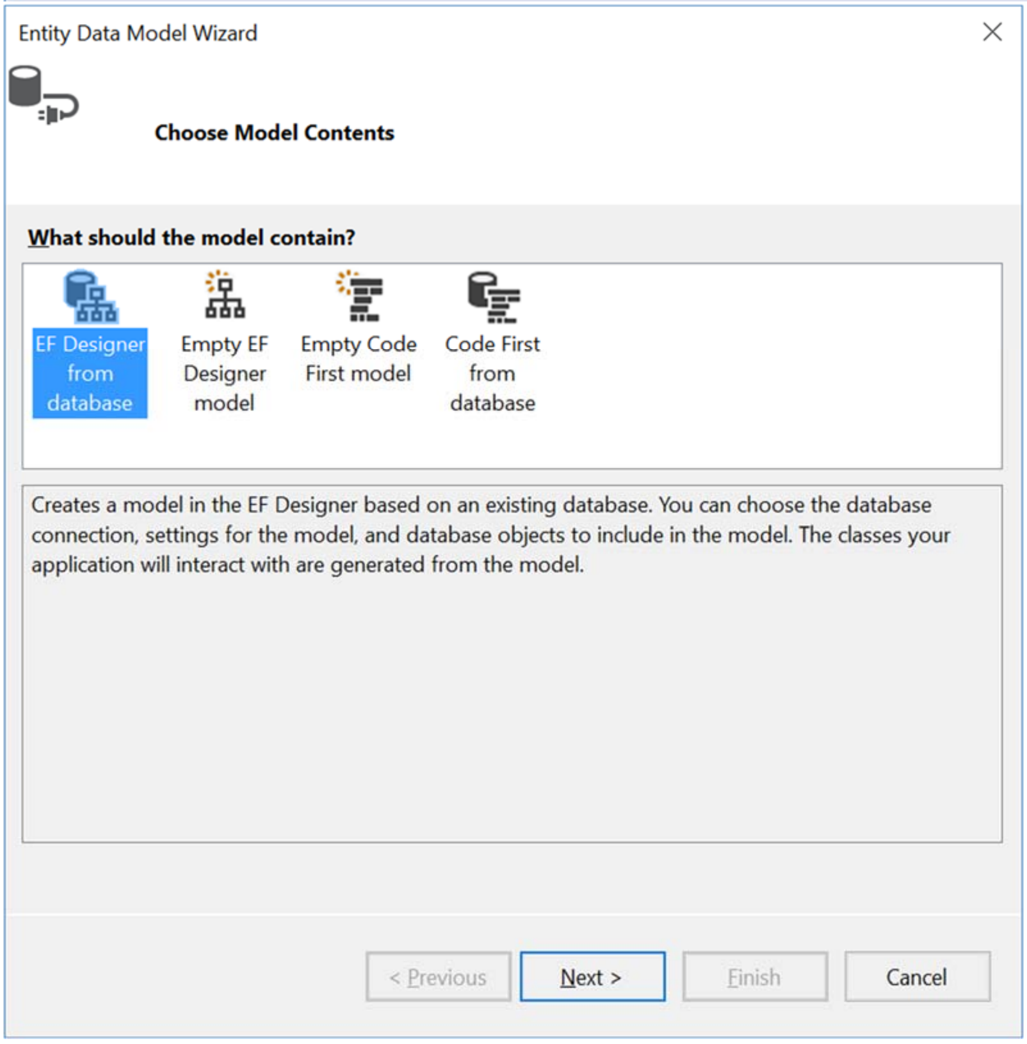

Add New Item - OnlineGame.Data

ADO.NET Entity Data Model	Visual C# Items
DataSet	Visual C# Items
EF 5.x DbContext Generator	Visual C# Items
EF 6.x DbContext Generator	Visual C# Items
LINQ to SQL Classes	Visual C# Items
Service-based Database	Visual C# Items
XML File	Visual C# Items
XML Schema	Visual C# Items

Type: Visual C# Items

A project item for creating an ADO.NET Entity Data Model.

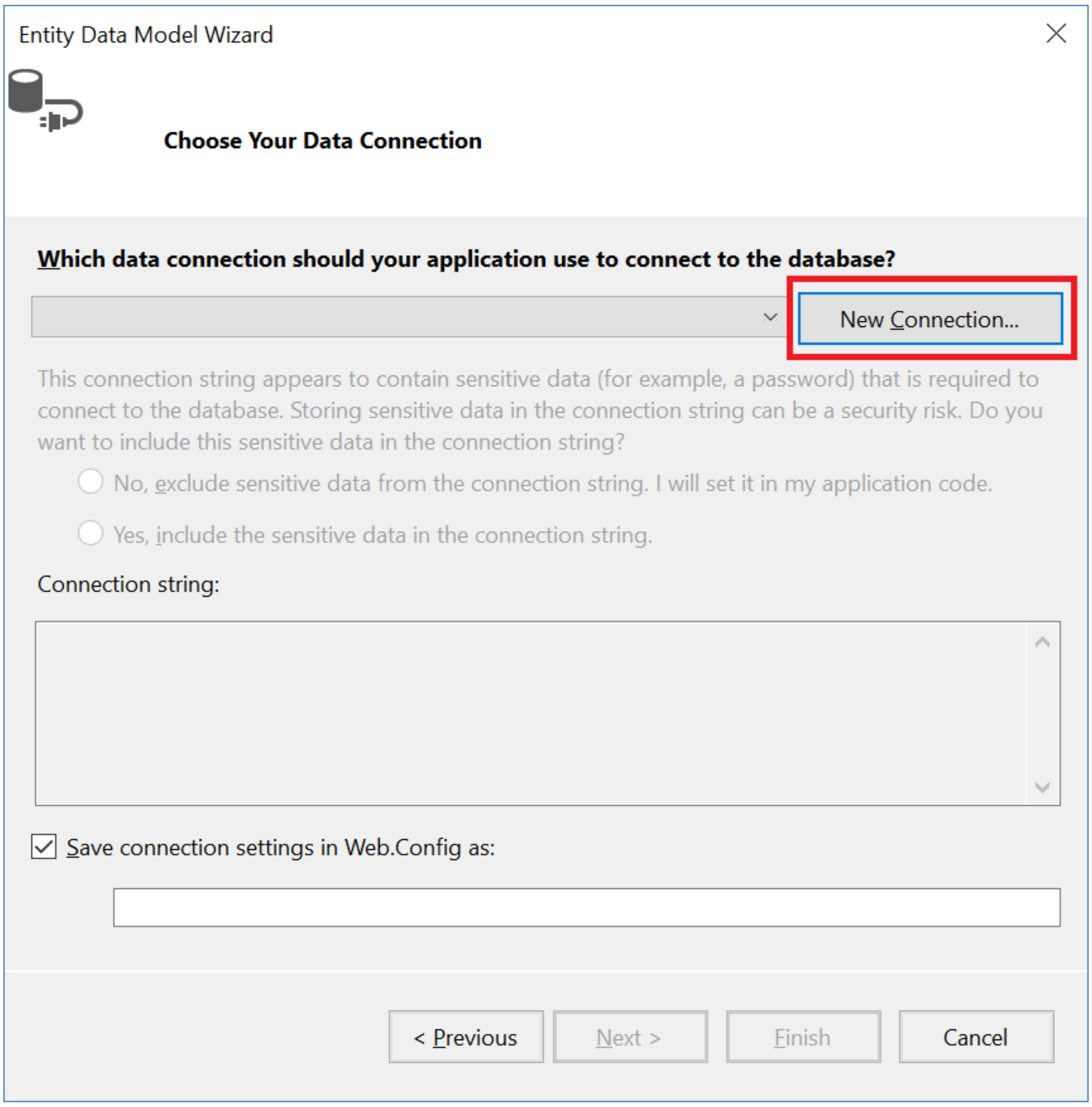
Name: OnlineGameDataModel


Entity Data Model Wizard

Choose Model Contents

What should the model contain?

Creates a model in the EF Designer based on an existing database. You can choose the database connection, settings for the model, and database objects to include in the model. The classes your application will interact with are generated from the model.



Entity Data Model Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

New Connection...

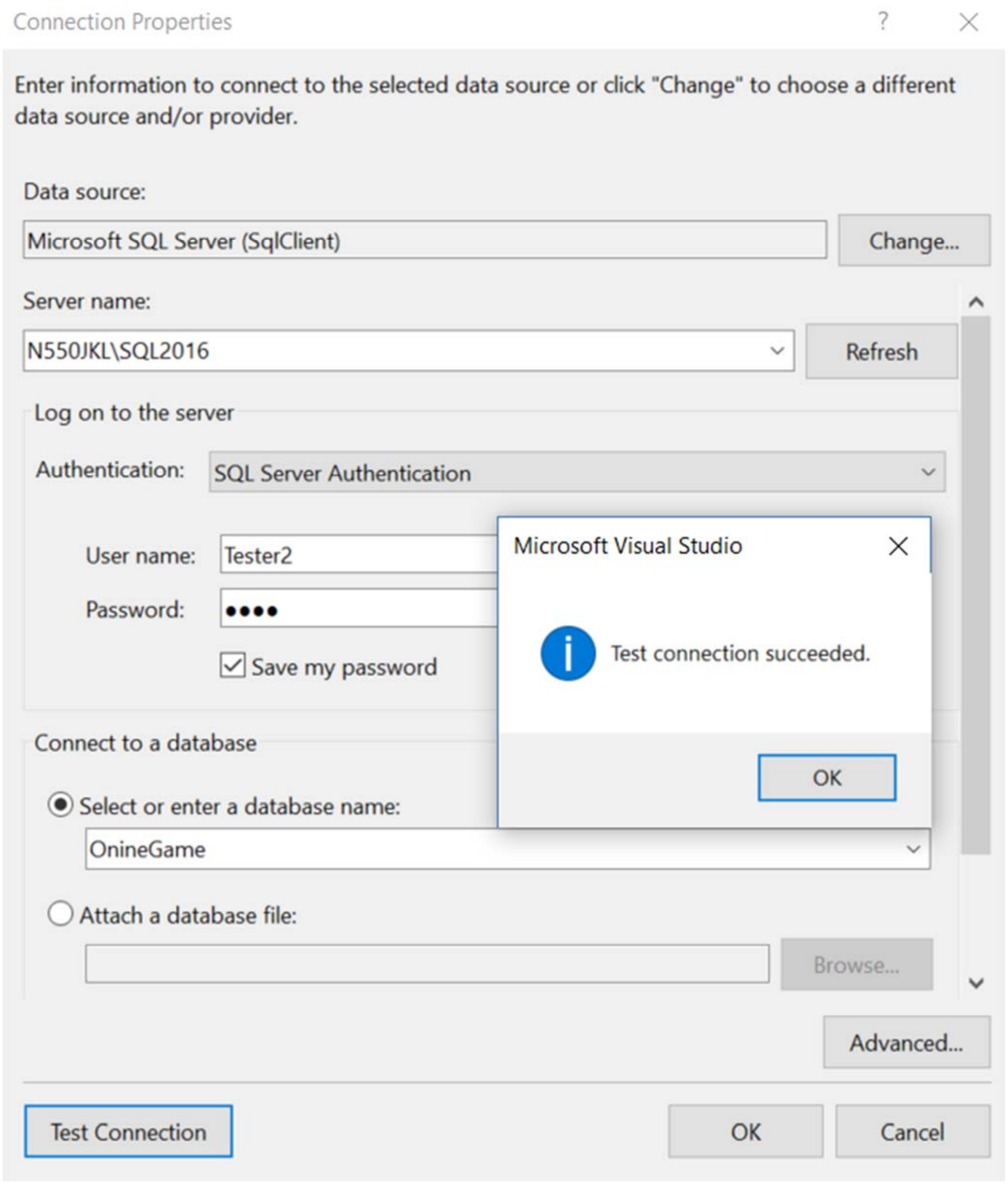
This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

- No, exclude sensitive data from the connection string. I will set it in my application code.
- Yes, include the sensitive data in the connection string.

Connection string:

[✓] Save connection settings in Web.Config as:

< Previous | Next > | Finish | Cancel


Connection Properties

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: Microsoft SQL Server (SqlClient)

Server name: N550JKL\SQL2016

Log on to the server

Authentication: SQL Server Authentication

User name: Tester2

Password: ●●●●

[✓] Save my password

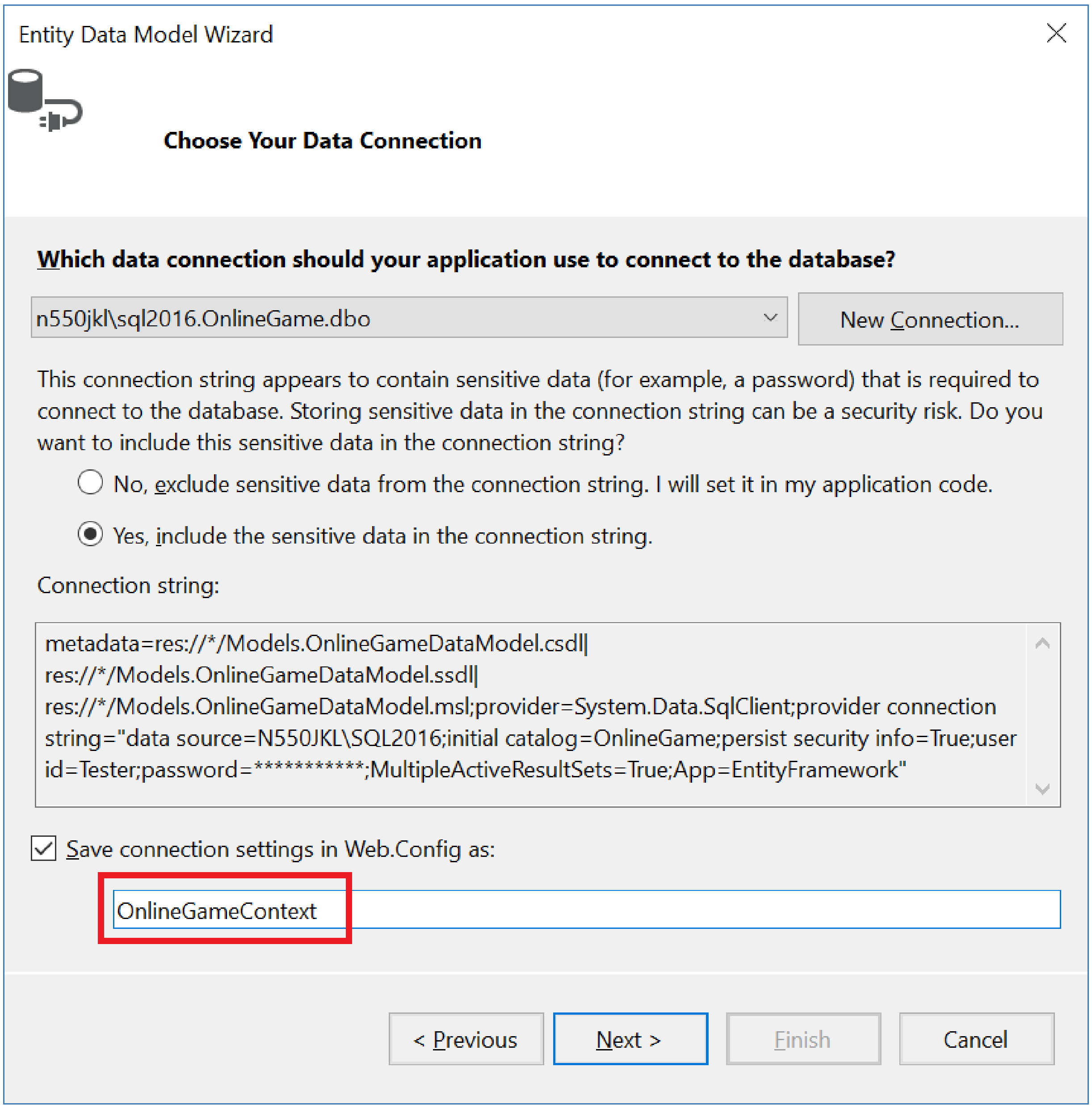
Connect to a database

(●) Select or enter a database name: OnineGame

( ) Attach a database file:

Microsoft Visual Studio

Test connection succeeded.


Entity Data Model Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

n550jkl\sql2016.OnlineGame.dbo

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

○ No, exclude sensitive data from the connection string. I will set it in my application code.

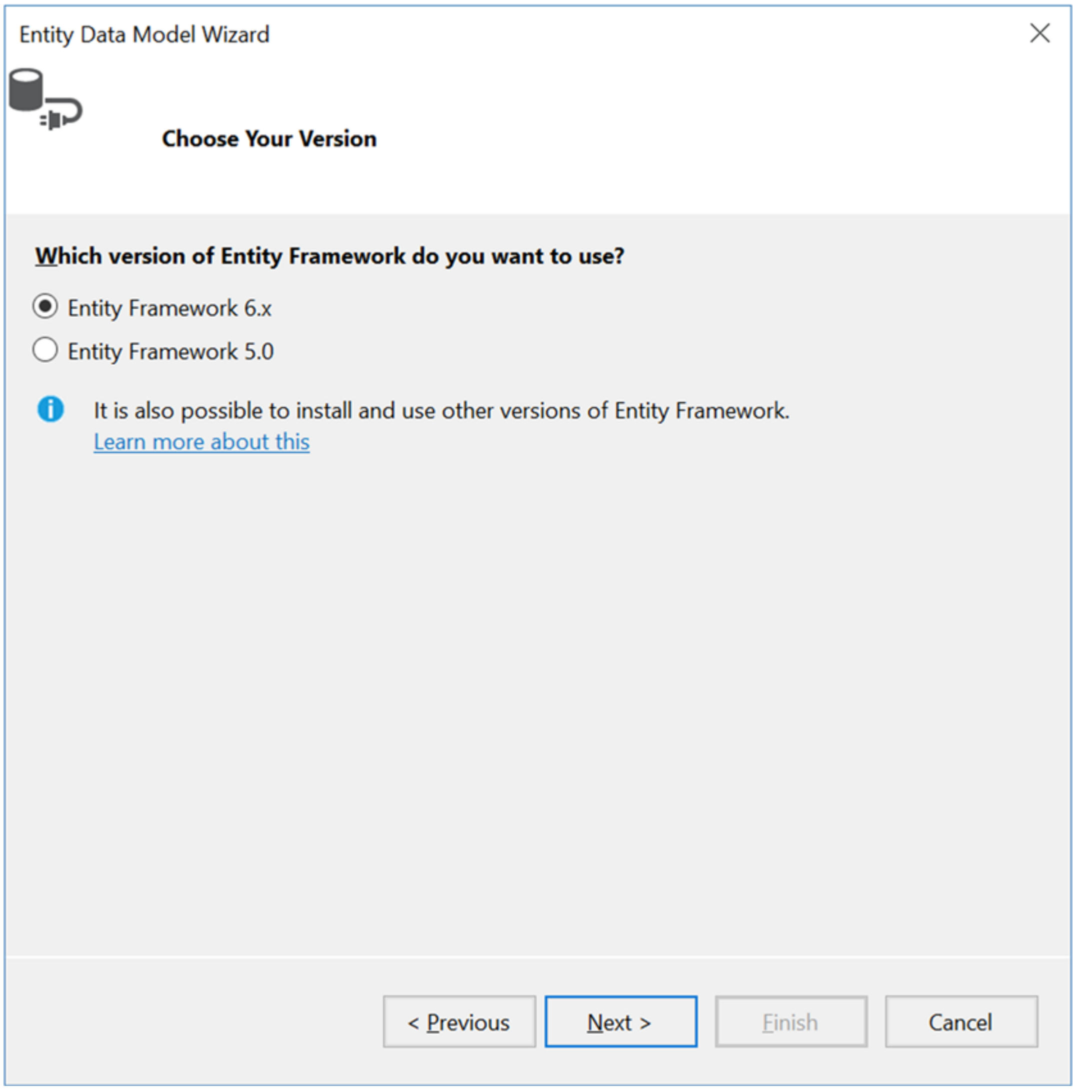
● Yes, include the sensitive data in the connection string.

Connection string:

```
metadata=res://*/Models.OnlineGameDataModel.csdl|
res://*/Models.OnlineGameDataModel.ssdl|
res://*/Models.OnlineGameDataModel.msl;provider=System.Data.SqlClient;provider connection
string="data source=N550JKL\SQL2016;initial catalog=OnlineGame;persist security info=True;user
id=Tester;password=***********;MultipleActiveResultSets=True;App=EntityFramework"
```

✓ Save connection settings in Web.Config as:

OnlineGameContext


Entity Data Model Wizard

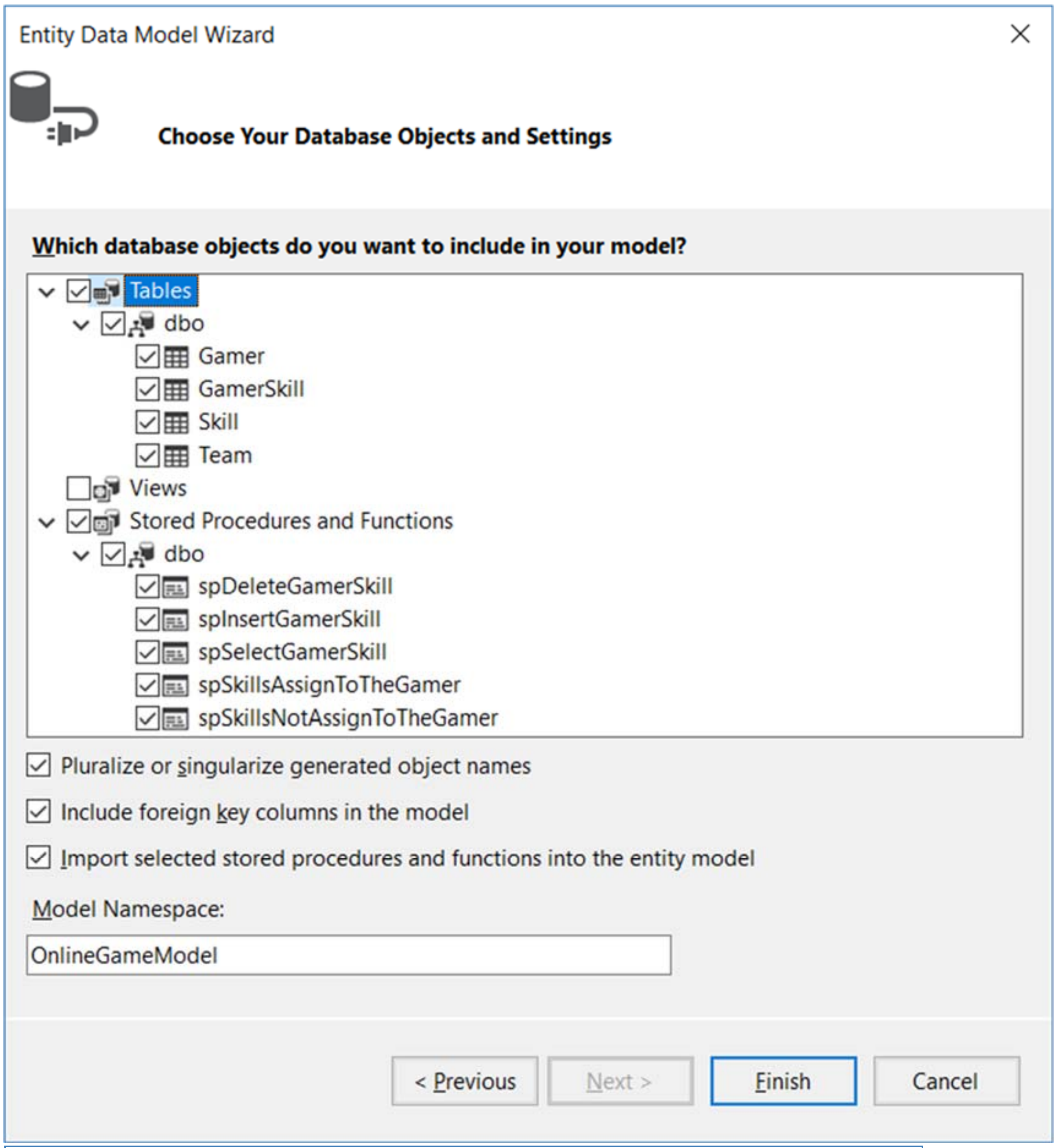
Choose Your Version

Which version of Entity Framework do you want to use?

[✓ Entity Framework 6.x]

Entity Framework 5.0

It is also possible to install and use other versions of Entity Framework.
Learn more about this


Entity Data Model Wizard

Choose Your Database Objects and Settings

Which database objects do you want to include in your model?

- [✓] Tables
  - [✓] dbo
    - [✓] Gamer
    - [✓] GamerSkill
    - [✓] Skill
    - [✓] Team
- [ ] Views
- [✓] Stored Procedures and Functions
  - [✓] dbo
    - [✓] spDeleteGamerSkill
    - [✓] spInsertGamerSkill
    - [✓] spSelectGamerSkill
    - [✓] spSkillsAssignToTheGamer
    - [✓] spSkillsNotAssignToTheGamer

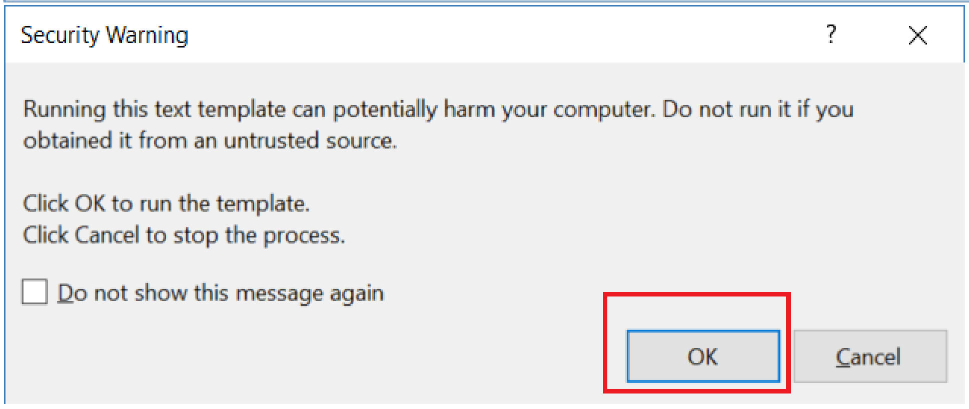
[✓] Pluralize or singularize generated object names

[✓] Include foreign key columns in the model

[✓] Import selected stored procedures and functions into the entity model

Model Namespace:

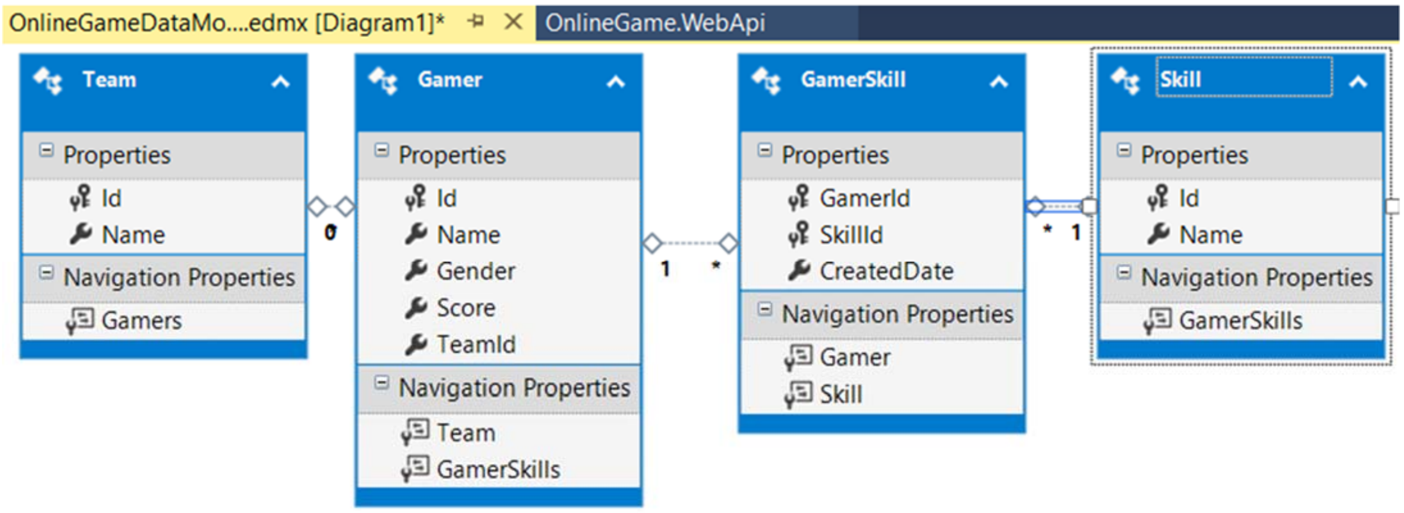
OnlineGameModel


Security Warning

Running this text template can potentially harm your computer. Do not run it if you obtained it from an untrusted source.

Click OK to run the template.
Click Cancel to stop the process.

[ ] Do not show this message again



4. OnlineGame.WebApi - API Controller

4.1. OnlineGame.WebApi/App_Start/WebApiConfig.cs - JSON Formatter

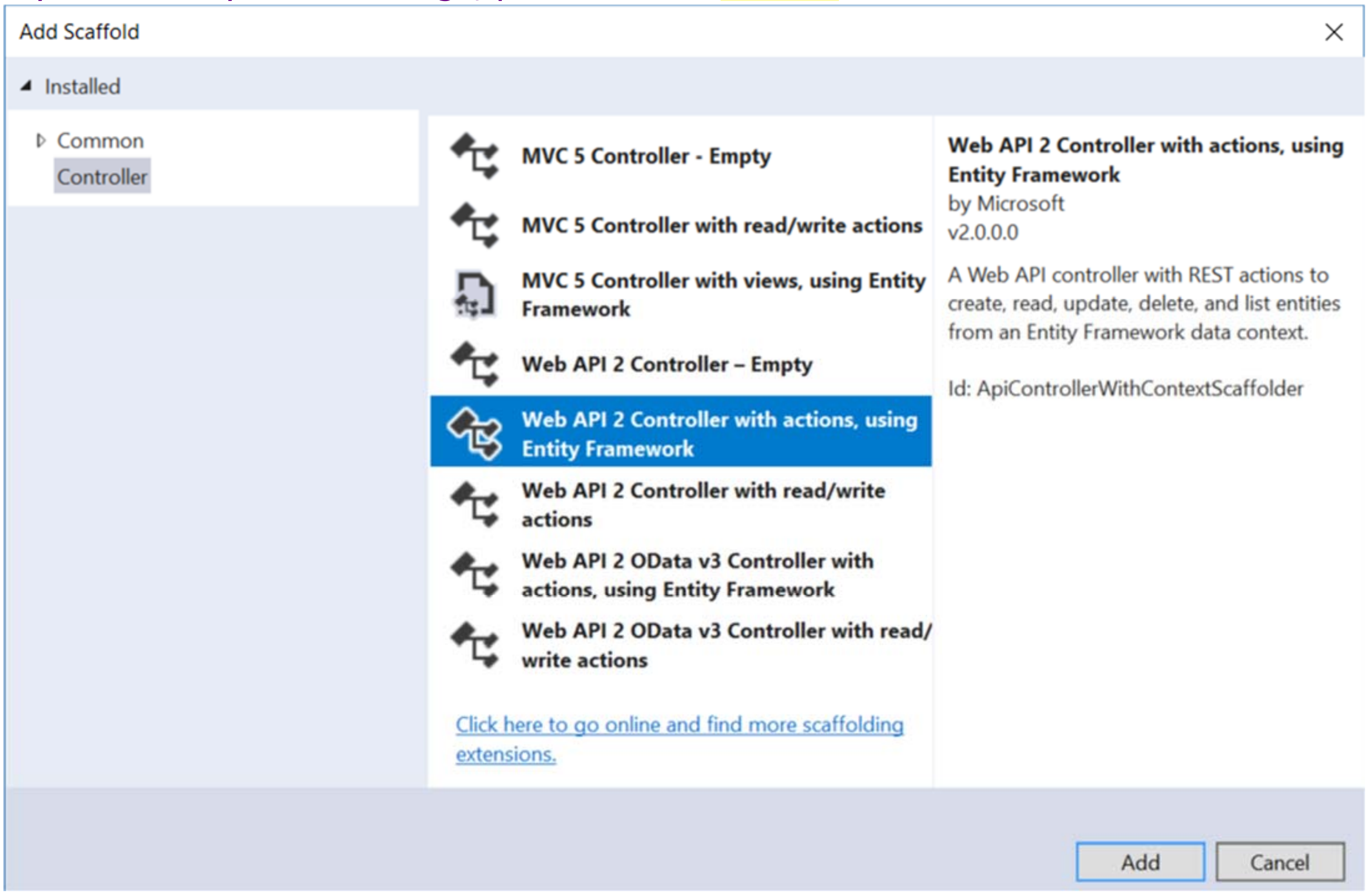
```
using System;
using System.Collections.Generic;
using System.Linq;
using  System.Net.Http.Formatting;
using System.Web.Http;
namespace OnlineGame.WebApi
{
    public static class WebApiConfig
    {
        public static void Register(HttpConfiguration config)
        {
            // Web API configuration and services
            // Web API routes
            config.MapHttpAttributeRoutes();
            config.Routes.MapHttpRoute(
                name: "DefaultApi",
                routeTemplate: "api/{controller}/{id}",
                defaults: new { id = RouteParameter.Optional }
            );

            //Use JSON formatter as a PreserveReferencesHandling.
            JsonMediaTypeFormatter json = config.Formatters.JsonFormatter;
            json.SerializerSettings.PreserveReferencesHandling =
Newtonsoft.Json.PreserveReferencesHandling.Objects;
            //Remove Xml Formatter
            config.Formatters.Remove(config.Formatters.XmlFormatter);
        }
    }
}

/*
//JsonMediaTypeFormatter json = config.Formatters.JsonFormatter;
//json.SerializerSettings.PreserveReferencesHandling =
Newtonsoft.Json.PreserveReferencesHandling.Objects;
//config.Formatters.Remove(config.Formatters.XmlFormatter);
```

```
Use JSON formatter as a PreserveReferencesHandling.
Remove Xml Formatter
Reference:
A.
https://forums.asp.net/t/1983286.aspx?Web+API+error+The+ObjectContent+1+type+failed+to+serialize+the+resp
onse+body+for+content+type+application+xml+charset+utf+8+
B.
https://stackoverflow.com/questions/23098191/failed-to-serialize-the-response-in-web-api-with-json
*/
```

4.2. OnlineGame.WebApi/Controllers/Api/GamerController.cs - Attribute routing

4.2.1. OnlineGame.WebApi/Controllers/Api/GamerController.cs - Attribute routing

Controllers/Api folder --> Right Click --> Add --> Controller
--> **Web API 2 Controller with actions, using Entity Framework**
--> **GamerController**
if you have any error message, please ensure re-build whole solutions.


Add Scaffold

Installed
- Common
- Controller

- MVC 5 Controller - Empty
- MVC 5 Controller with read/write actions
- MVC 5 Controller with views, using Entity Framework
- Web API 2 Controller – Empty
- Web API 2 Controller with actions, using Entity Framework
- Web API 2 Controller with read/write actions
- Web API 2 OData v3 Controller with actions, using Entity Framework
- Web API 2 OData v3 Controller with read/write actions

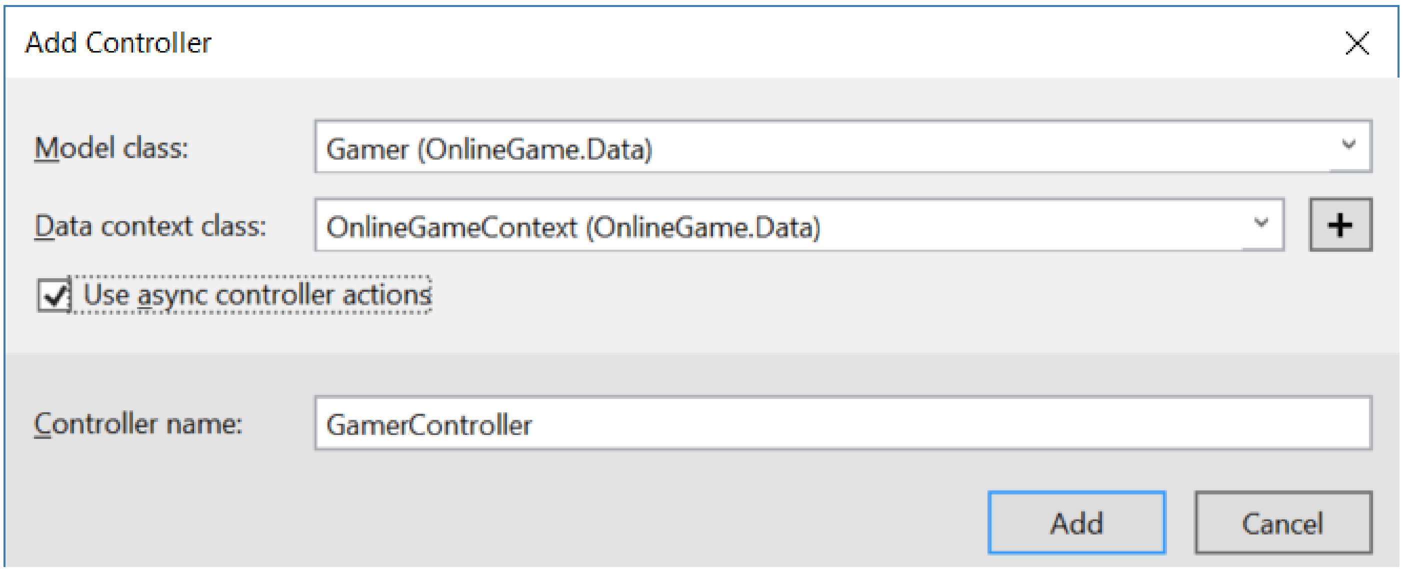
Click here to go online and find more scaffolding extensions.

Web API 2 Controller with actions, using Entity Framework
by Microsoft
v2.0.0.0

A Web API controller with REST actions to create, read, update, delete, and list entities from an Entity Framework data context.

Id: ApiControllerWithContextScaffolder

Add Cancel


Add Controller

Model class: Gamer (OnlineGame.Data)

Data context class: OnlineGameContext (OnlineGame.Data)

[✓] Use async controller actions

Controller name: GamerController

4.2.2. OnlineGame.WebApi/Controllers/Api/GamerController.cs - Attribute routing

```
using System.Collections.Generic;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Threading.Tasks;
using System.Web.Http;
using System.Web.Http.Description;
using OnlineGame.WebApi.Models;
namespace OnlineGame.WebApi.Controllers.Api
{
    public class GamerController : ApiController
    {
        private OnlineGameContext _db = new OnlineGameContext();

        // GET: api/Gamer
        [HttpGet]
        public async Task<IEnumerable<Gamer>> GetGamers()
        {
            return await _db.Gamers.ToListAsync();
        }

        // GET: api/Gamer/1
        //Convention-based routing.
        [HttpGet]
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamer(int id)
        {
            Gamer gamer = await _db.Gamers.FindAsync(id);
            if (gamer == null) return NotFound();  //404
            return Ok(gamer);   //200
        }

        [HttpGet]
        //Attribute Routing
        [Route("api/gamer/{id}/skills")]    // GET: api/gamer/1/skills
```

```
public async Task<IHttpActionResult> GetGamerSkills(int id)
 {
     Gamer gamer = await _db.Gamers.FindAsync(id);
    if (gamer == null) return NotFound();  //404
     List<Skill> skills = await GetSkillsByGamerId(id);
    return Ok(skills);   //200
 }

 [HttpGet]
 [Route("api/gamer/skills/{id}")]    // GET: api/gamer/skills/1
public async Task<IHttpActionResult> GetGamerSkills2(int id)
 {
     Gamer gamer = await _db.Gamers.FindAsync(id);
    if (gamer == null) return NotFound();  //404
     List<Skill> skills = await GetSkillsByGamerId(id);
    return Ok(skills);   //200
 }

// PUT: api/Gamer/1
 [HttpPut]
 [ResponseType(typeof(void))]
public async Task<IHttpActionResult> PutGamer(int id, Gamer gamer)
 {
    if (!ModelState.IsValid) return BadRequest(ModelState);  //400
    //if (id != gamer.Id)   return BadRequest();

    //1.
     gamer.Id = id;
     _db.Entry(gamer).State = EntityState.Modified;  //update the gamer

    //2.
    //Gamer currentGamer = await _db.Gamers.FirstOrDefaultAsync(g => g.Id == id);
    //if (currentGamer == null) return NotFound();  //404
    //currentGamer.Name = gamer.Name;
    //currentGamer.Gender = gamer.Gender;
    //currentGamer.Score = gamer.Score;
    //currentGamer.GameMoney = gamer.GameMoney;

    try
     {
        await _db.SaveChangesAsync();
        return Ok();    //200
     }
    catch (DbUpdateConcurrencyException)
     {
        if (!GamerExists(id)) return NotFound();  //404
        throw;
     }
 }

// POST: api/Gamer
 [HttpPost]
 [ResponseType(typeof(Gamer))]
public async Task<IHttpActionResult> PostGamer(Gamer gamer)
 {
    if (!ModelState.IsValid) return BadRequest(ModelState); //400
     _db.Gamers.Add(gamer);
```

```
            await _db.SaveChangesAsync();
            //Return Created/201.
            return CreatedAtRoute("DefaultApi", new { id = gamer.Id }, gamer);    //Created/201
        }

        // DELETE: api/Gamer/1
        [HttpDelete]
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> DeleteGamer(int id)
        {
            Gamer gamer = await _db.Gamers.FindAsync(id);
            if (gamer == null) return NotFound();   //404
            _db.Gamers.Remove(gamer);
            await _db.SaveChangesAsync();
            return Ok(gamer);   //200
        }

        private async Task<List<Skill>> GetSkillsByGamerId(int gamerId)
        {
            IQueryable<GamerSkill> gamerSkills = _db.Gamers
                .SelectMany(
                    g => g.GamerSkills, //The source of gamerSkill in second parameter
                    (g, gamerSkill) =>
                        new { GamerId = g.Id, GamerSkill = gamerSkill }) //Projection to a anonymous type
                .Where(gs => gs.GamerId == gamerId) //gamer id==gamerId
                .Select(gs => gs.GamerSkill); // Projection to GamerSkill Type

            List<Skill> skills =
                    await gamerSkills.Select(gamerSkill => _db.Skills.FirstOrDefault(s => s.Id ==
gamerSkill.SkillId)).ToListAsync();

            //Projection to Skill
            return skills;
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing) _db.Dispose();   //Dispose DBContext
            base.Dispose(disposing);
        }

        private bool GamerExists(int id)
        {
            return _db.Gamers.Count(e => e.Id == id) > 0;
        }
    }
}

/*
1.
1.1.
By default, the HTTP verb GET maps to a method that has the name Get() or "Get" prefix.
E.g. Get(), GetGamers, GetXXX()
If you want the HTTP verb GET maps to the method name without "Get" prefix.
You can use [HttpGet] attribute.
1.2.
[HttpGet] attribute maps HTTP verb GET.
[HttpPost] attribute maps HTTP verb POST.
[HttpPut] attribute maps HTTP verb PUT.
```

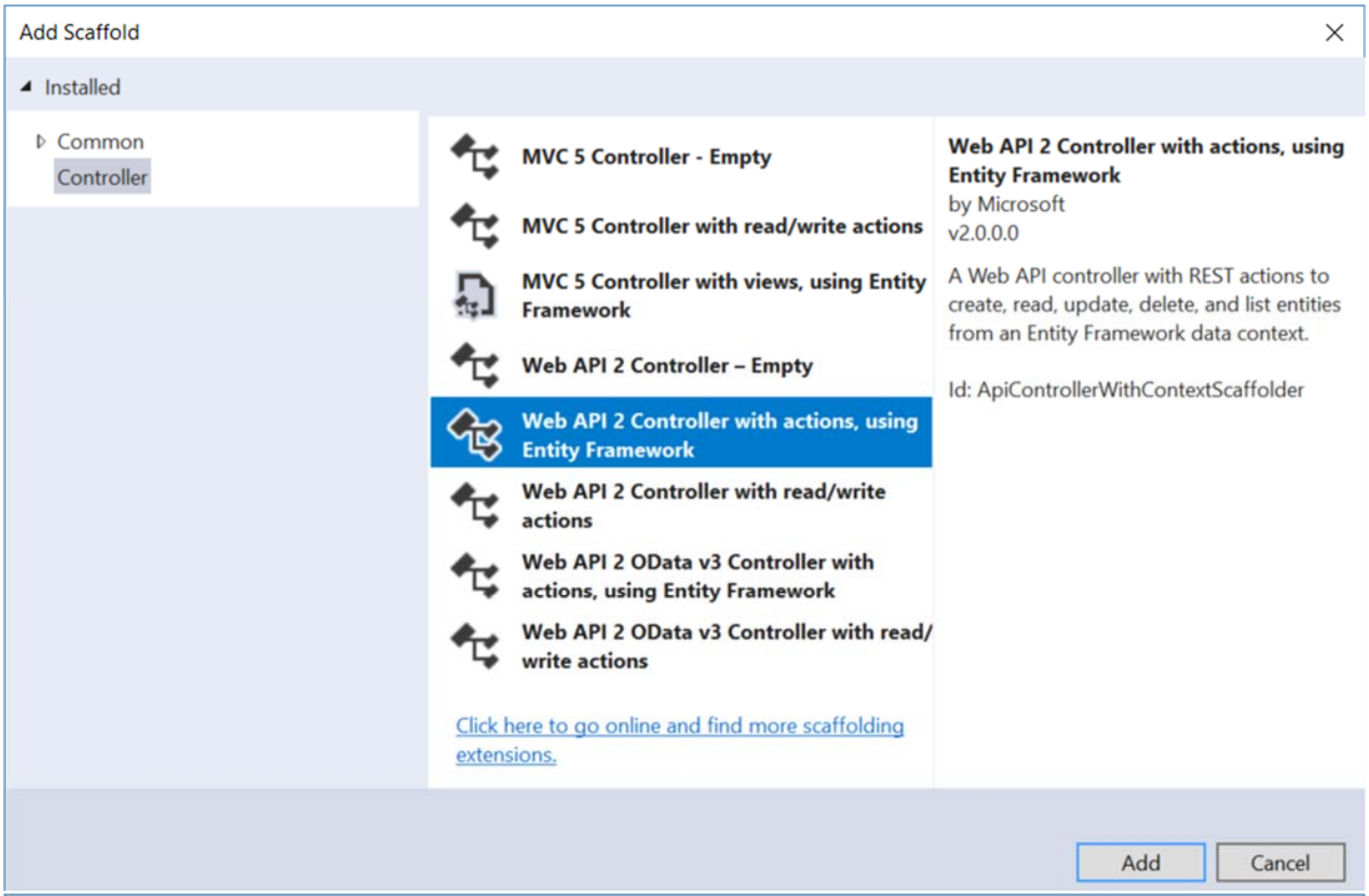
```
[HttpDelete] attribute maps HTTP verb DELETE.
----------------------------
2.
[FromUri] V.S. [FromBody]
Web Api default binding parameter convention
2.1.
By default, if the parameter is a simple type,
Web Api will try to get value from uri.
E.g. int, double, bool, ...etc.
2.2.
By default, if the parameter is a complex type,
Web Api will try to get value from the request body.
E.g. Gamer
-----------------
2.3.
//[HttpPut]
//public async Task<IHttpActionResult> UpdateGamer(int id, Gamer gamer)
By Default, the Web Api will try to get id from uri, and gamer from request body as below code.
//[HttpPut]
//public async Task<IHttpActionResult> UpdateGamer([FromUri]int id, [FromBody]Gamer gamer)
E.g.
A.
PUT
http://localhost:58302/api/Gamer/8
B.
Request Header
Host: localhost:58302
Content-Type: application/json
B.1.
Accept: application/json
means we request JSON format response.
B.2.
Content-Type: application/json
The client will post a data to the server, the data format is JSON
C.
Request Body
{
"Name":"NameEight XYZ222",
"Gender":"Male",
"Score":450,
"GameMoney":1500
}
-----------------
2.4.
//[HttpPut]
//public async Task<IHttpActionResult> UpdateGamer([FromBody]int id, [FromUri]Gamer gamer)
[FromBody] will enfroce to get id from request body
[FromUri] will enforce to get gamer from uri
E.g.
A.
PUT
http://localhost:58302/api/Gamer?Name=NameEight%20XYZ333&Gender=Male&Score=450&GameMoney=1500
B.
Request Header
Host: localhost:58302
Content-Type: application/json
B.1.
Accept: application/json
means we request JSON format response.
B.2.
Content-Type: application/json
The client will post a data to the server, the data format is JSON
C.
Request Body
8
----------------------------
6.
```

```
Attribute routing
-----------------
6.1.
E.g.
//public async Task<IHttpActionResult> GetGamer(int id){...}
....
//[Route("api/gamer/{id}/skills")]
//public async Task<IHttpActionResult> GetGamerSkills(int id){...}
When we call "api/gamer/1" and if we don't have Route attribute,
the API will be confused,
because both GetGamerSkills() and GetGamer() can map to "api/gamer/1".
Thus, we need Route attribute
[Route("api/gamer/{id}/skills")] will make GetGamerSkills() map to something like "api/gamer/1/skills".
Thus, GetGamer() can map to something like "api/gamer/1".
-----------------
6.2.
In this case,
GetGamer() is using Convention-based routing.
GetGamerSkills() is using Attribute Routing.
-----------------
6.3.
In
OnlineGame.WebApi/WebApiConfig.cs/WebApiConfig.cs
//config.MapHttpAttributeRoutes();
It enables Attribute Routing.
*/
```

4.3. OnlineGame.WebApi/Controllers/Api/GamerTwoController.cs - RoutePrefix and Route attribute

4.3.1. OnlineGame.WebApi/Controllers/Api/GamerTwoController.cs
- RoutePrefix and Route attribute

Controllers/Api folder --> Right Click --> Add --> Controller
--> **Web API 2 Controller with actions, using Entity Framework**
--> **GamerTwoController**
if you have any error message, please ensure re-build whole solutions.


Add Scaffold

Installed

Common

Controller

MVC 5 Controller - Empty

MVC 5 Controller with read/write actions

MVC 5 Controller with views, using Entity Framework

Web API 2 Controller – Empty

Web API 2 Controller with actions, using Entity Framework

Web API 2 Controller with read/write actions

Web API 2 OData v3 Controller with actions, using Entity Framework

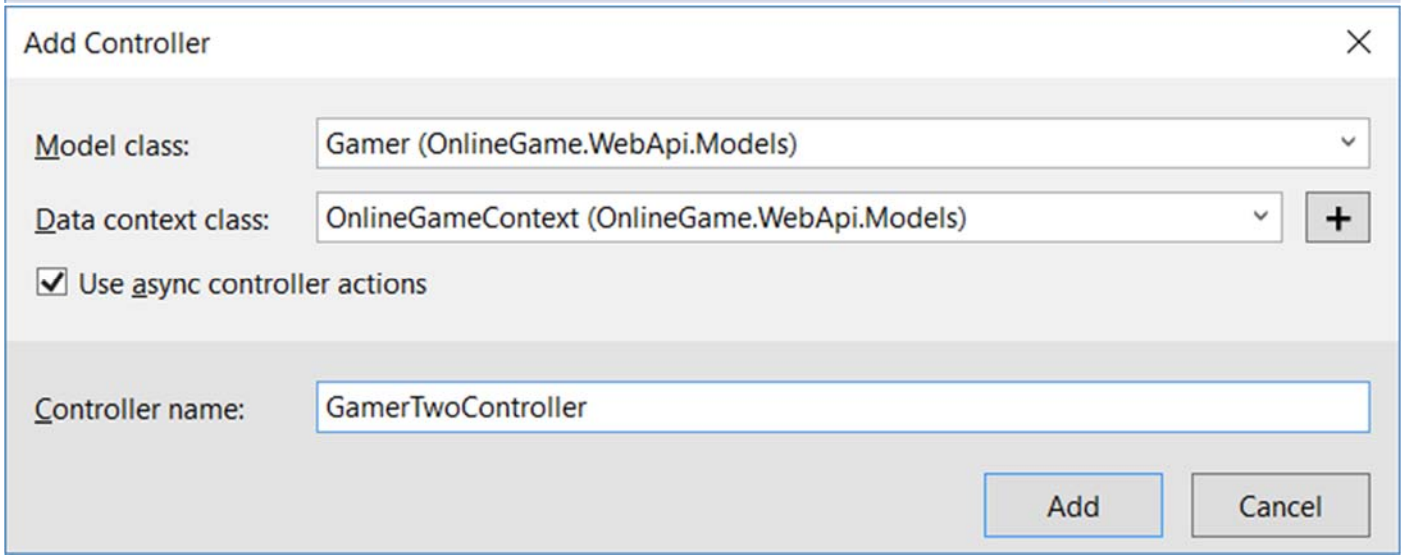
Web API 2 OData v3 Controller with read/write actions

Click here to go online and find more scaffolding extensions.

Web API 2 Controller with actions, using Entity Framework
by Microsoft
v2.0.0.0

A Web API controller with REST actions to create, read, update, delete, and list entities from an Entity Framework data context.

Id: ApiControllerWithContextScaffolder


Add Controller

Model class: Gamer (OnlineGame.WebApi.Models)

Data context class: OnlineGameContext (OnlineGame.WebApi.Models)

☑ Use async controller actions

Controller name: GamerTwoController

4.3.2. OnlineGame.WebApi/Controllers/Api/GamerTwoController.cs - RoutePrefix and Route attribute

```
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Threading.Tasks;
using System.Web.Http;
using System.Web.Http.Description;
using OnlineGame.WebApi.Models;

namespace OnlineGame.WebApi.Controllers.Api
{
```

```csharp
[RoutePrefix("api/gamer2")]
public class GamerTwoController : ApiController
{
    private OnlineGameContext _db = new OnlineGameContext();

    // GET: api/Gamertwo
    public IQueryable<Gamer> GetGamers()
    {
        return _db.Gamers;
    }

    // GET: api/Gamer2
    [Route("")]
    public IQueryable<Gamer> GetGamers2()
    {
        return _db.Gamers;
    }

    // GET: api/gamer2/api/gamer2
    [Route("api/gamer2")]
    public IQueryable<Gamer> GetGamers3()
    {
        return _db.Gamers;
    }

    // GET: api/gamer2/api/getGamers
    [Route("api/getGamers")]
    public IQueryable<Gamer> GetGamers4()
    {
        return _db.Gamers;
    }

    // GET: api/getGamers
    [Route("~/api/getGamers")]
    public IQueryable<Gamer> GetGamers5()
    {
        return _db.Gamers;
    }

    // GET: api/gamerTwo/1
    [ResponseType(typeof(Gamer))]
    public async Task<IHttpActionResult> GetGamer(int id)
    {
         Gamer gamer = await _db.Gamers.FindAsync(id);
        if (gamer == null) return NotFound();  //404
        return Ok(gamer);   //200
    }

    // GET: api/gamer2/1
    [Route("{id}")]
    [ResponseType(typeof(Gamer))]
    public async Task<IHttpActionResult> GetGamer2(int id)
    {
         Gamer gamer = await _db.Gamers.FindAsync(id);
        if (gamer == null) return NotFound();  //404
        return Ok(gamer);   //200
    }
```

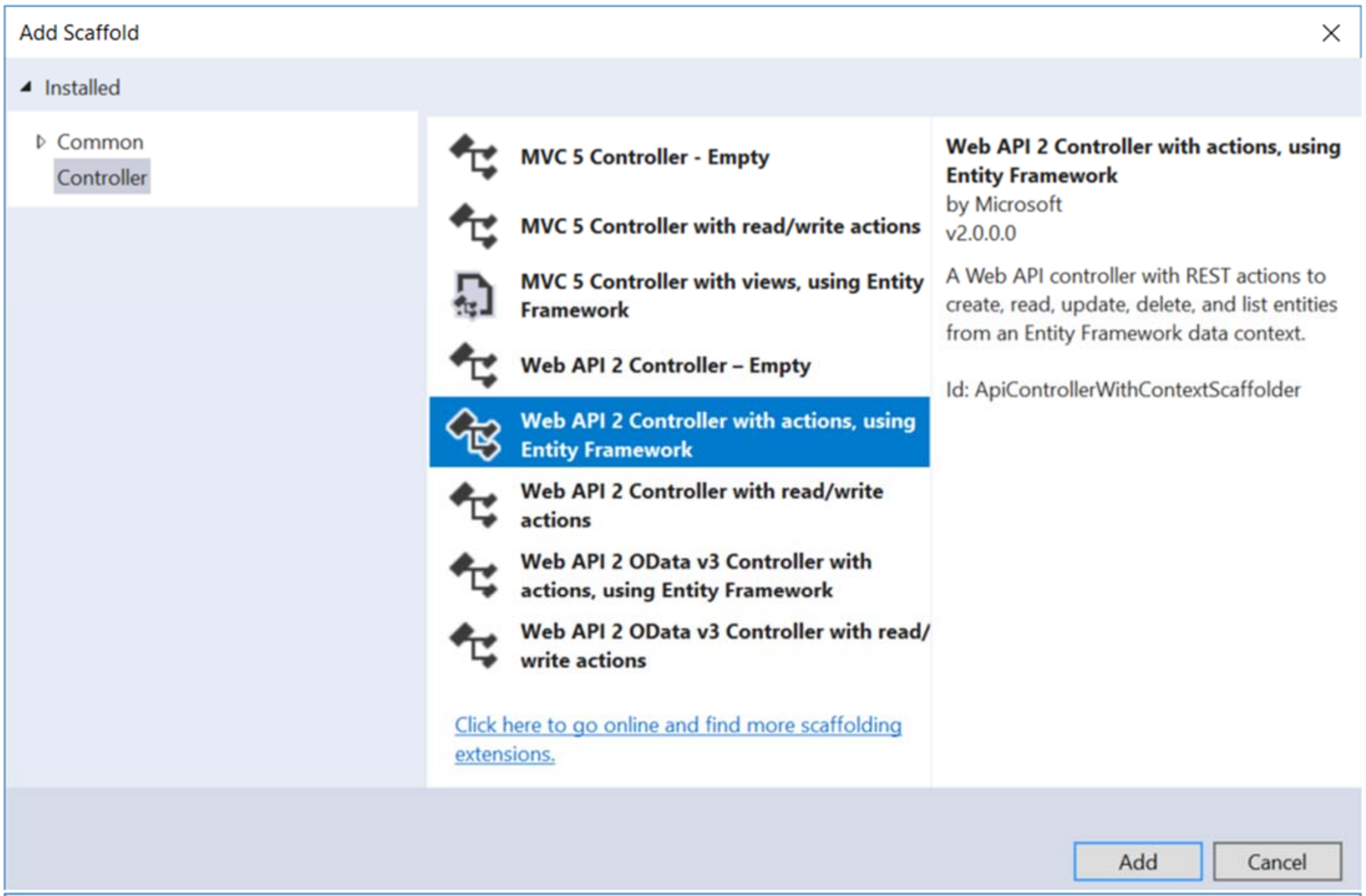
```
// GET: api/gamer2/api/gamer2/1
 [ResponseType(typeof(Gamer))]
 [Route("api/gamer2/{id}")]
public async Task<IHttpActionResult> GetGamer3(int id)
 {
     Gamer gamer = await _db.Gamers.FindAsync(id);
    if (gamer == null) return NotFound();  //404
    return Ok(gamer);   //200
 }

// GET: api/gamer2GetGamerById/1
 [ResponseType(typeof(Gamer))]
 [Route("~/api/gamer2GetGamerById/{id}")]
public async Task<IHttpActionResult> GetGamer4(int id)
 {
     Gamer gamer = await _db.Gamers.FindAsync(id);
    if (gamer == null) return NotFound();  //404
    return Ok(gamer);   //200
 }

 [HttpGet]
 [Route("api/gamer2/{gamerId}/skills")]    // GET: api/gamer2/api/gamer2/1/skills
public async Task<IHttpActionResult> GetGamerSkills(int gamerId)
 {
     Gamer gamer = await _db.Gamers.FindAsync(gamerId);
    if (gamer == null) return NotFound();  //404
     List<Skill> skills = await GetSkillsByGamerId(gamerId);
    return Ok(skills);   //200
 }

 [HttpGet]
 [Route("api/gamer2/skills/{gamerId}")]    // GET: api/gamer2/api/gamer2/skills/1
public async Task<IHttpActionResult> GetGamerSkills2(int gamerId)
 {
     Gamer gamer = await _db.Gamers.FindAsync(gamerId);
    if (gamer == null) return NotFound();  //404
     List<Skill> skills = await GetSkillsByGamerId(gamerId);
    return Ok(skills);   //200
 }

 [HttpGet]
 [Route("skills/{gamerId}")]    // GET: api/gamer2/skills/1
public async Task<IHttpActionResult> GetGamerSkills3(int gamerId)
 {
     Gamer gamer = await _db.Gamers.FindAsync(gamerId);
    if (gamer == null) return NotFound();  //404
     List<Skill> skills = await GetSkillsByGamerId(gamerId);
    return Ok(skills);   //200
 }

 [HttpGet]
 [Route("~/api/getGamerSkillsByGamerId/{gamerId}")]    // GET: api/getGamerSkillsByGamerId/1
public async Task<IHttpActionResult> GetGamerSkills4(int gamerId)
 {
     Gamer gamer = await _db.Gamers.FindAsync(gamerId);
    if (gamer == null) return NotFound();  //404
     List<Skill> skills = await GetSkillsByGamerId(gamerId);
```

```
            return Ok(skills);   //200
        }

        private async Task<List<Skill>> GetSkillsByGamerId(int gamerId)
        {
            IQueryable<GamerSkill> gamerSkills = _db.Gamers
                .SelectMany(
                    g => g.GamerSkills, //The source of gamerSkill in second parameter
                    (g, gamerSkill) =>
                        new { GamerId = g.Id, GamerSkill = gamerSkill }) //Projection to a anonymous type
                .Where(gs => gs.GamerId == gamerId) //gamer id==gamerId
                .Select(gs => gs.GamerSkill); // Projection to GamerSkill Type

            List<Skill> skills =
                    await gamerSkills.Select(gamerSkill => _db.Skills.FirstOrDefault(s => s.Id ==
gamerSkill.SkillId)).ToListAsync();

            //Projection to Skill
            return skills;
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing) _db.Dispose();   //Dispose DBContext
            base.Dispose(disposing);
        }

        private bool GamerExists(int id)
        {
            return _db.Gamers.Count(e => e.Id == id) > 0;
        }
    }
}

/*
7.
RoutePrefix and Route attribute
//[RoutePrefix("api/gamer2")]
RoutePrefix attribute is for route prefix at the controller level.
Route attribute use that route prefix plus its own route value.
//[Route("~/api/getGamerSkillsByGamerId/{gamerId}")]
if you want to  override the route prefix,
just use ~ (tilde) symbol
*/
```

4.4. OnlineGame.WebApi/Controllers/Api/GamerThreeController.cs - attribute routing constraints

4.4.1. OnlineGame.WebApi/Controllers/Api/GamerThreeController.cs - attribute routing constraints

Controllers/Api folder --> Right Click --> Add --> Controller
--> **Web API 2 Controller with actions, using Entity Framework**
--> **GamerThreeController**
if you have any error message, please ensure re-build whole solutions.


Add Scaffold

Installed

Common

Controller

MVC 5 Controller - Empty

MVC 5 Controller with read/write actions

MVC 5 Controller with views, using Entity Framework

Web API 2 Controller – Empty

Web API 2 Controller with actions, using Entity Framework

Web API 2 Controller with read/write actions

Web API 2 OData v3 Controller with actions, using Entity Framework

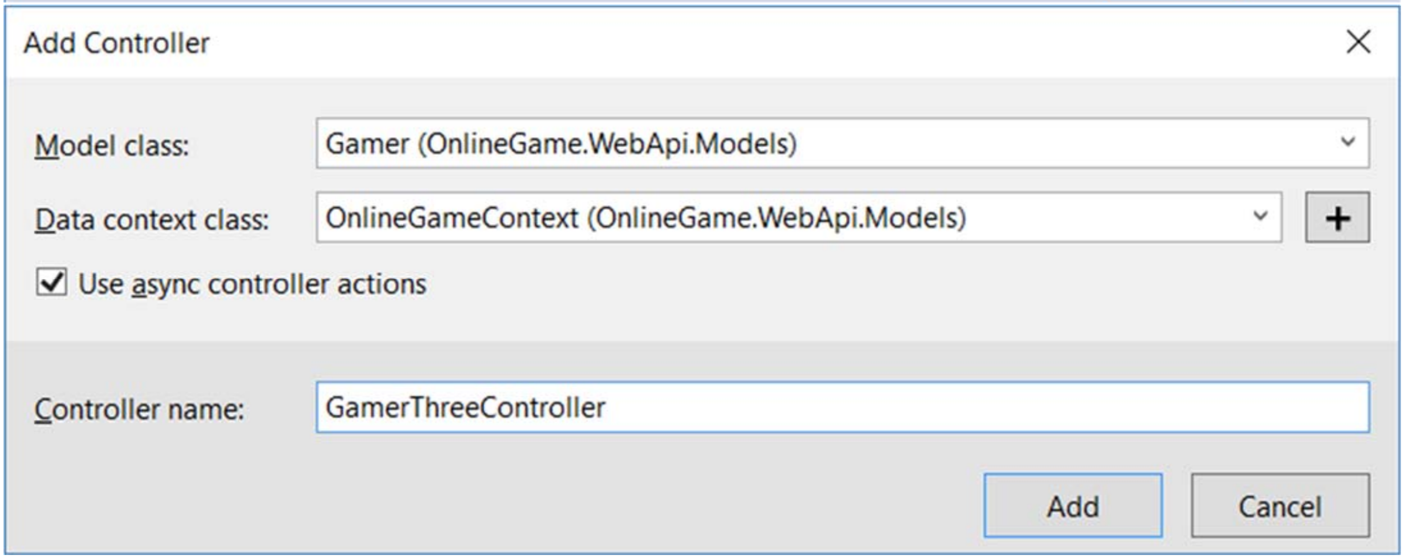
Web API 2 OData v3 Controller with read/write actions

Click here to go online and find more scaffolding extensions.

Web API 2 Controller with actions, using Entity Framework
by Microsoft
v2.0.0.0

A Web API controller with REST actions to create, read, update, delete, and list entities from an Entity Framework data context.

Id: ApiControllerWithContextScaffolder


Add Controller

Model class: Gamer (OnlineGame.WebApi.Models)

Data context class: OnlineGameContext (OnlineGame.WebApi.Models)

[✓] Use async controller actions

Controller name: GamerThreeController

4.4.2. OnlineGame.WebApi/Controllers/Api/GamerThreeController.cs - attribute routing constraints

```
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Threading.Tasks;
using System.Web.Http;
using System.Web.Http.Description;
using OnlineGame.WebApi.Models;

namespace OnlineGame.WebApi.Controllers.Api
{
```

```csharp
[RoutePrefix("api/gamer3")]
public class GamerThreeController : ApiController
{
    private OnlineGameContext _db = new OnlineGameContext();

    // GET: api/GamerThree
    public IQueryable<Gamer> GetGamers()
    {
        return _db.Gamers;
    }

    // GET: api/GamerThree/1
    [ResponseType(typeof(Gamer))]
    public async Task<IHttpActionResult> GetGamer(int id)
    {
        Gamer gamer = await _db.Gamers.FindAsync(id);
        if (gamer == null) return NotFound();   //404
        return Ok(gamer);
    }

    // GET: api/gamer3/GetGamerBySomething/2
    [Route("GetGamerBySomething/{gamerId:int}")]
    public async Task<IHttpActionResult> GetGamerBySomething(int gamerId)
    {
        Gamer gamer = await _db.Gamers.FindAsync(gamerId);
        if (gamer == null) return NotFound();   //404
        return Ok(gamer);
    }

    // GET: api/gamer3/GetGamerBySomething/male
    //[Route("GetGamerBySomething/{gender:string}")]    //Error, string type is not valid
    [Route("GetGamerBySomething/{gender:alpha}")]
    //alpha means uppercase or lowercase alphabet.
    public async Task<IHttpActionResult> GetGamerBySomething(string gender)
    {
        List<Gamer> gamer =
            await _db.Gamers.Where(
                g => g.Gender.ToLower().Equals(gender.ToLower()))   //it is not case sensitive
                .ToListAsync();
        return Ok(gamer);
    }

    [ResponseType(typeof(Gamer))]
    [Route("{gamerId}")]    // GET: api/gamer3/1
    public async Task<IHttpActionResult> GetGamer2(int gamerId)
    {
        Gamer gamer = await _db.Gamers.FindAsync(gamerId);
        if (gamer == null) return NotFound();   //404
        return Ok(gamer);
    }

    // GET: api/gamer3/getGamerById/1
    // gamerId must be int and min is 2
    [Route("getGamerById/{gamerId:int:min(2)}")]
    [ResponseType(typeof(Gamer))]
    public async Task<IHttpActionResult> GetGamerById(int gamerId)
    {
```

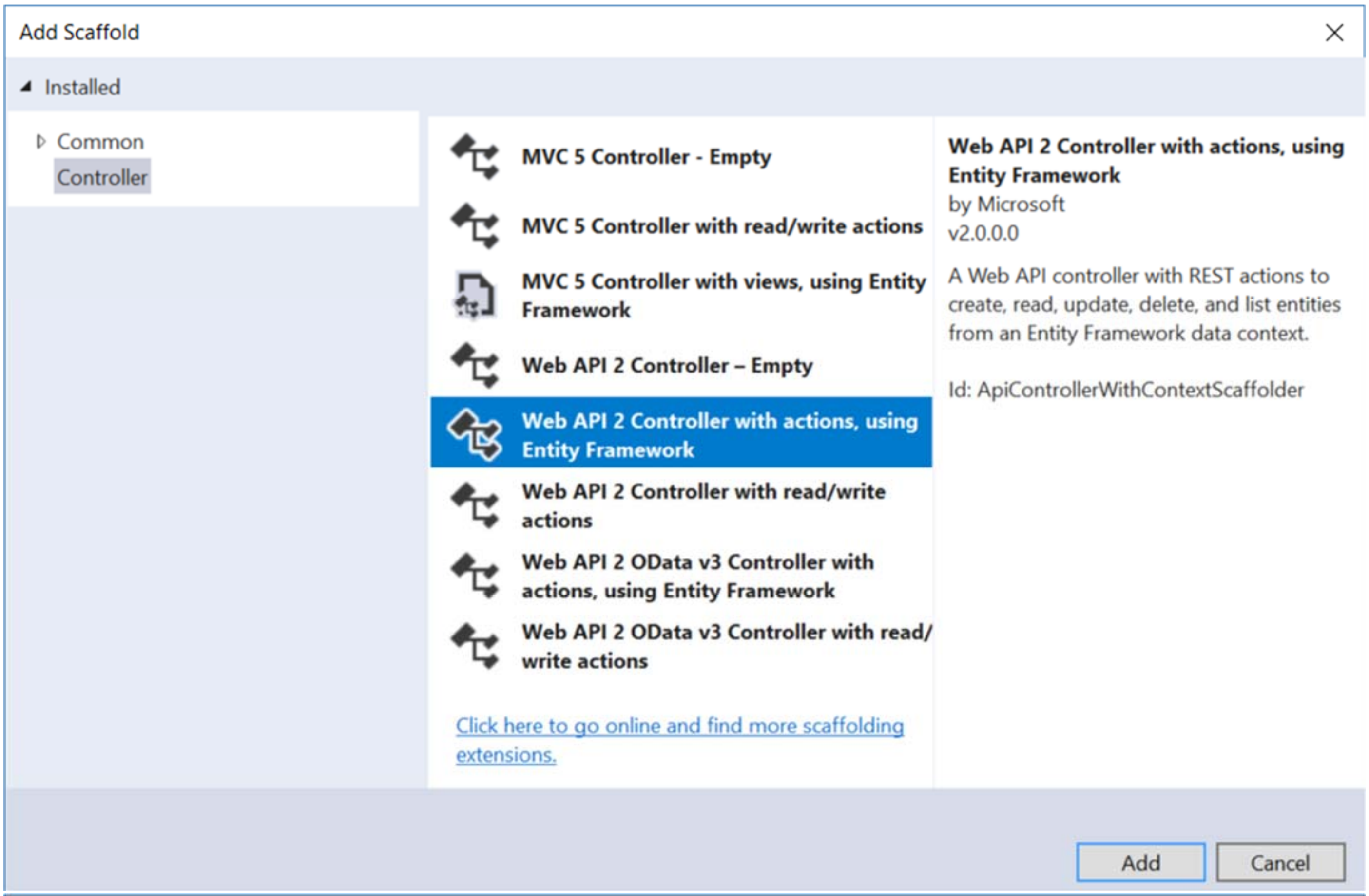
```
             Gamer gamer = await _db.Gamers.FindAsync(gamerId);
            if (gamer == null) return NotFound();   //404
            return Ok(gamer);
        }

        // GET: api/gamer3/getGamerById2/1
        // gamerId must be int and min is 2, max is 5
        [Route("getGamerById2/{gamerId:int:min(2):max(5)}")]
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamerById2(int gamerId)
        {
             Gamer gamer = await _db.Gamers.FindAsync(gamerId);
            if (gamer == null) return NotFound();   //404
            return Ok(gamer);
        }

        // GET: api/gamer3/getGamerById3/1
        // gamerId must be int and min is 2, max is 5
        [Route("getGamerById3/{gamerId:range(2,5)}")]
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamerById3(int gamerId)
        {
             Gamer gamer = await _db.Gamers.FindAsync(gamerId);
            if (gamer == null) return NotFound();   //404
            return Ok(gamer);
        }

        // GET: api/gamer3/getGamersByGender/female
        //[Route("getGamersByGender/{gender:string}")]    //Error, string type is not valid
        [Route("getGamersByGender/{gender:alpha}")] //alpha means uppercase or lowercase alphabet
characters.
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamersByGender(string gender)
        {
            List<Gamer> gamer =
                await _db.Gamers.Where(
                    g => g.Gender.ToLower().Equals(gender.ToLower()))   //it is not case sensitive
                    .ToListAsync();
            return Ok(gamer);
        }

        // GET: api/gamer3/getGamersByGender2/female     //will return nothing, it is case sensitive
        // GET: api/gamer3/getGamersByGender2/Female
        [Route("getGamersByGender2/{gender:alpha}")]
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamersByGender2(string gender)
        {
            List<Gamer> gamer =
                await _db.Gamers.Where(
                    g => g.Gender.Equals(gender))   //it is case sensitive
                    .ToListAsync();
            return Ok(gamer);
        }

        // GET: api/gamer3/getGamersByGender3/female        //404
        // GET: api/gamer3/getGamersByGender3/male
        [Route("getGamersByGender3/{gender:alpha:maxlength(5)}")]
        //alpha means uppercase or lowercase alphabet characters.
        //max alpha length is 5
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamersByGender3(string gender)
        {
            List<Gamer> gamer =
                await _db.Gamers.Where(
                    g => g.Gender.ToLower().Equals(gender.ToLower()))
                    .ToListAsync();
            return Ok(gamer);
        }

        // GET: api/gamer3/getGamersByGender4/female
        // GET: api/gamer3/getGamersByGender4/male      //404
        [Route("getGamersByGender4/{gender:alpha:minlength(5):maxlength(7)}")]
        //alpha means uppercase or lowercase alphabet characters.
        //max alpha length is 7, and min length is 5.
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamersByGender4(string gender)
        {
            List<Gamer> gamer =
                await _db.Gamers.Where(
                    g => g.Gender.ToLower().Equals(gender.ToLower()))   //it is not case sensitive
                    .ToListAsync();
            return Ok(gamer);
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing) _db.Dispose();
            base.Dispose(disposing);
        }

        private bool GamerExists(int id)
        {
            return _db.Gamers.Count(e => e.Id == id) > 0;
        }
    }
}

/*
8.
attribute routing constraints
Reference:
https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/attribute-routing-in-web-api-2#route-constraints
Routing constraints can apply to decimal, double, float, long, bool...etc.
--------------
8.1.
//// GET: api/gamer3/GetGamerBySomething/2
//[Route("GetGamerBySomething/{gamerId:int}")]
//public async Task<IHttpActionResult> GetGamerBySomething(int gamerId)
int means integer
...
//// GET: api/gamer3/GetGamerBySomething/male
////[Route("GetGamerBySomething/{gender:string}")]    //Error, string type is not valid
//[Route("GetGamerBySomething/{gender:alpha}")]
//public async Task<IHttpActionResult> GetGamerBySomething(string gender)
alpha means uppercase or lowercase alphabet.
--------------
8.2.
```

```
//[Route("getGamerById/{gamerId:int:min(2)}")]
//public async Task<IHttpActionResult> GetGamerById(int gamerId)
GET: api/gamer3/getGamerById/1
gamerId must be int and min is 2
--------------
8.3.
//[Route("getGamerById2/{gamerId:int:min(2):max(5)}")]
//public async Task<IHttpActionResult> GetGamerById2(int gamerId)
GET: api/gamer3/getGamerById2/1
gamerId must be int and min is 2, max is 5
--------------
8.4.
//[Route("getGamerById3/{gamerId:range(2,5)}")]
//public async Task<IHttpActionResult> GetGamerById3(int gamerId)
GET: api/gamer3/getGamerById3/1
gamerId must be int and min is 2, max is 5
--------------
8.5.
////[Route("getGamersByGender/{gender:string}")]    //Error, string type is not valid
//[Route("getGamersByGender/{gender:alpha}")]
//public async Task<IHttpActionResult> GetGamersByGender(string gender)
alpha means uppercase or lowercase alphabet characters.
GET: api/gamer3/getGamersByGender/female
--------------
8.7.
//[Route("getGamersByGender3/{gender:alpha:maxlength(5)}")]
//public async Task<IHttpActionResult> GetGamersByGender3(string gender)
GET: api/gamer3/getGamersByGender3/female        //404
GET: api/gamer3/getGamersByGender3/male
alpha means uppercase or lowercase alphabet characters.
max alpha length is 5
--------------
8.8.
//[Route("getGamersByGender3/{gender:alpha:maxlength(5)}")]
//public async Task<IHttpActionResult> GetGamersByGender3(string gender)
GET: api/gamer3/getGamersByGender3/female        //404
GET: api/gamer3/getGamersByGender3/male
alpha means uppercase or lowercase alphabet characters.
max alpha length is 5
--------------
8.9.
//[Route("getGamersByGender4/{gender:alpha:minlength(5):maxlength(7)}")]
//public async Task<IHttpActionResult> GetGamersByGender4(string gender)
GET: api/gamer3/getGamersByGender4/female
GET: api/gamer3/getGamersByGender4/male      //404
alpha means uppercase or lowercase alphabet characters.
max alpha length is 7, and min length is 5.
*/
```

4.5. OnlineGame.WebApi/Controllers/Api/GamerFourController.cs - Route names

4.5.1. OnlineGame.WebApi/Controllers/Api/GamerFourController.cs - Route names

Controllers/Api folder --> Right Click --> Add --> Controller
--> **Web API 2 Controller with actions, using Entity Framework**
--> **GamerFourController**
if you have any error message, please ensure re-build whole solutions.


Add Scaffold

Installed

Common

Controller

MVC 5 Controller - Empty

MVC 5 Controller with read/write actions

MVC 5 Controller with views, using Entity Framework

Web API 2 Controller – Empty

Web API 2 Controller with actions, using Entity Framework

Web API 2 Controller with read/write actions

Web API 2 OData v3 Controller with actions, using Entity Framework

Web API 2 OData v3 Controller with read/write actions

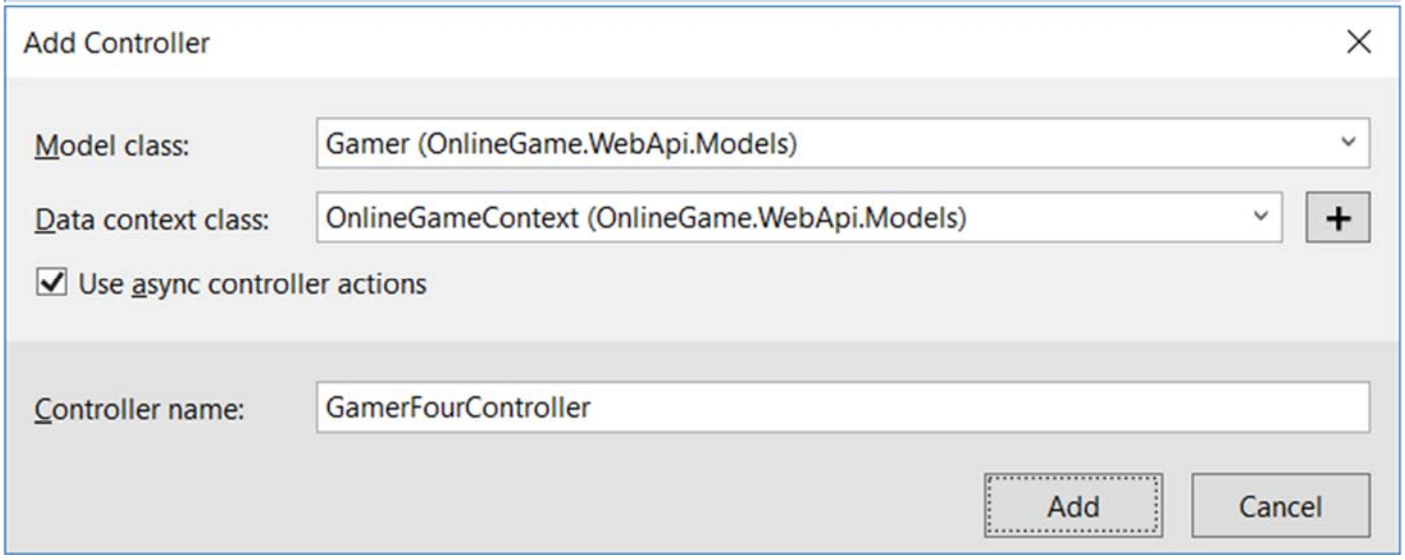
Click here to go online and find more scaffolding extensions.

Web API 2 Controller with actions, using Entity Framework
by Microsoft
v2.0.0.0

A Web API controller with REST actions to create, read, update, delete, and list entities from an Entity Framework data context.

Id: ApiControllerWithContextScaffolder

Add Cancel


Add Controller

Model class: Gamer (OnlineGame.WebApi.Models)

Data context class: OnlineGameContext (OnlineGame.WebApi.Models)

☑ Use async controller actions

Controller name: GamerFourController

Add Cancel

4.5.2. OnlineGame.WebApi/Controllers/Api/GamerFourController.cs - Route names

```
using System;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Threading.Tasks;
using System.Web.Http;
using System.Web.Http.Description;
using OnlineGame.WebApi.Models;

namespace OnlineGame.WebApi.Controllers.Api
{
```

```
[RoutePrefix("api/gamer4")]
public class GamerFourController : ApiController
{
    private OnlineGameContext _db = new OnlineGameContext();

    // GET: api/GamerFour
    public IQueryable<Gamer> GetGamers()
    {
        return _db.Gamers;
    }

    // GET: api/GamerFour/1
    [ResponseType(typeof(Gamer))]
    public async Task<IHttpActionResult> GetGamer(int id)
    {
         Gamer gamer = await _db.Gamers.FindAsync(id);
        if (gamer == null) return NotFound();   //404
        return Ok(gamer);
    }

    // GET: api/Gamer4/1
    [ResponseType(typeof(Gamer))]
    [Route("{id:int}", Name = "GetGamerById")]
    public async Task<IHttpActionResult> GetGamerById(int id)
    {
         Gamer gamer = await _db.Gamers.FindAsync(id);
        if (gamer == null) return NotFound();   //404
        return Ok(gamer);
    }

    //IHttpActionResult is from Web API 2
    // POST: api/GamerFour
    [HttpPost]
    [ResponseType(typeof(Gamer))]
    public async Task<IHttpActionResult> PostGamer(Gamer gamer)
    {
        if (!ModelState.IsValid) return BadRequest(ModelState); //400
         _db.Gamers.Add(gamer);
        await _db.SaveChangesAsync();

        //Return Created/201.
        return CreatedAtRoute("DefaultApi", new { id = gamer.Id }, gamer);    //Created/201
    }

    // POST: api/Gamer4/AddGamer
    [HttpPost]
    [Route("AddGamer")]
    public async Task<HttpResponseMessage> AddGamer(Gamer gamer)
    {
        if (!ModelState.IsValid)
            return Request.CreateErrorResponse(HttpStatusCode.BadRequest,
               "ModelState is invalid");    //400

         _db.Gamers.Add(gamer);
        await _db.SaveChangesAsync();

        //Return Created/201.
```

```
     HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.Created);
     response.Headers.Location = new Uri(Request.RequestUri + "/" + gamer.Id);

    return response;    //Created/201
}

//IHttpActionResult is from Web API 2
// POST: api/Gamer4/AddGamer2
[Route("AddGamer2")]
[HttpPost]
[ResponseType(typeof(Gamer))]
public async Task<IHttpActionResult> AddGamer2(Gamer gamer)
{
    if (!ModelState.IsValid) return BadRequest(ModelState); //400
     _db.Gamers.Add(gamer);
    await _db.SaveChangesAsync();
    //Return Created/201.
    return CreatedAtRoute("DefaultApi", new { id = gamer.Id }, gamer);    //Created/201
}

// POST: api/Gamer4/AddGamer3
[HttpPost]
[Route("AddGamer3")]
public async Task<HttpResponseMessage> AddGamer3(Gamer gamer)
{
    if (!ModelState.IsValid)
        return Request.CreateErrorResponse(HttpStatusCode.BadRequest,
           "ModelState is invalid");    //400
     _db.Gamers.Add(gamer);
    await _db.SaveChangesAsync();

    //Return Created/201.
     HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.Created);
     response.Headers.Location = new
                 Uri(Url.Link("GetGamerById", new { id = gamer.Id }));

    return response;    //Created/201
}

// POST: api/Gamer4/AddGamer4
[HttpPost]
[Route("AddGamer4")]
public async Task<IHttpActionResult> AddGamer4(Gamer gamer)
{
    if (!ModelState.IsValid) return BadRequest(ModelState); //400
     _db.Gamers.Add(gamer);
    await _db.SaveChangesAsync();
    //Return Created/201.
    return CreatedAtRoute("GetGamerById", new { id = gamer.Id }, gamer);    //Created/201
}

protected override void Dispose(bool disposing)
{
    if (disposing) _db.Dispose();
    base.Dispose(disposing);
}
```

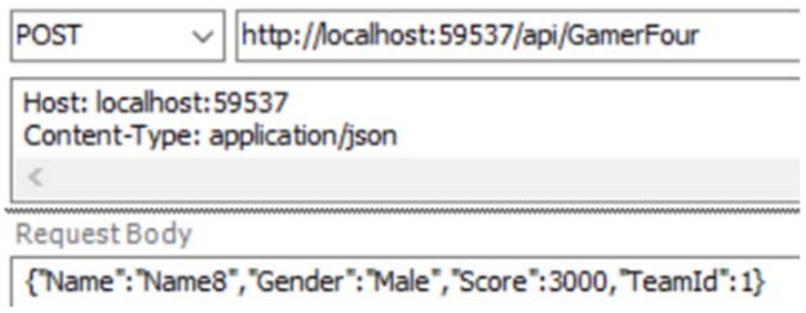
```
        private bool GamerExists(int id)
        {
            return _db.Gamers.Count(e => e.Id == id) > 0;
        }
    }
}

/*
9.
Route names
9.1.
E.g.
//[Route("{id:int}", Name = "GetGamerById")]
//public async Task<IHttpActionResult> GetGamerById(int id)
...
//HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.Created);
//response.Headers.Location = new
//    Uri(Url.Link("GetGamerById", new { id = gamer.Id }));
...
//return CreatedAtRoute("GetGamerById", new { id = gamer.Id }, gamer);    //Created/201
9.2.
//return CreatedAtRoute("DefaultApi", new { id = gamer.Id }, gamer);    //Created/201
...
//HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.Created);
//response.Headers.Location = new Uri(Request.RequestUri + "/" + gamer.Id);
*/
```

4.5.3. Post Request

4.5.3.1. Post Request - public async Task<IHttpActionResult> PostGamer(Gamer gamer)

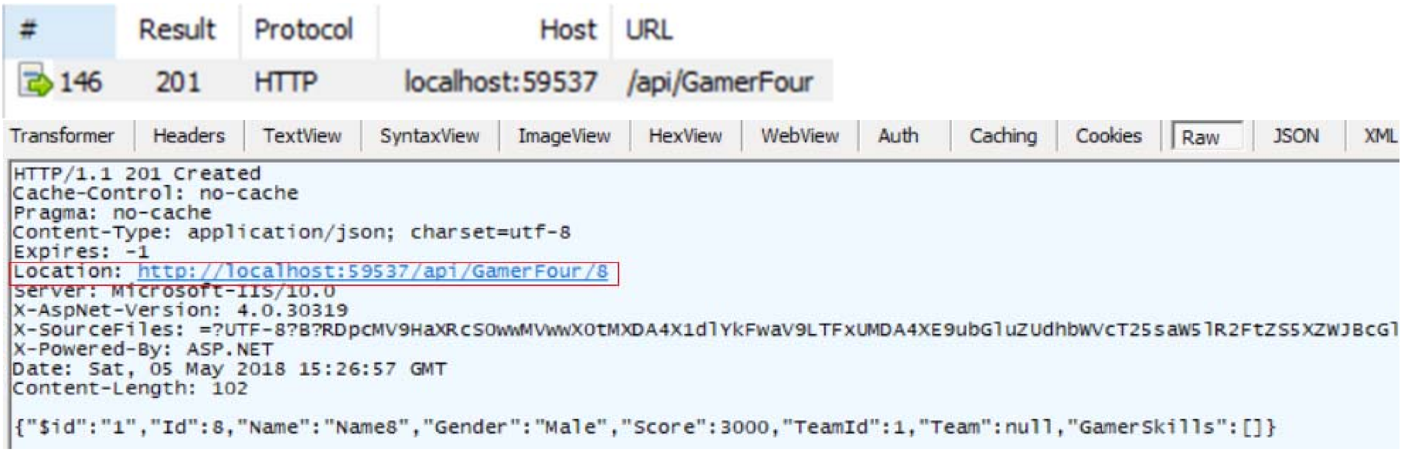
```
//IHttpActionResult is from Web API 2
// POST: api/GamerFour
[HttpPost]
[ResponseType(typeof(Gamer))]
public async Task<IHttpActionResult> PostGamer(Gamer gamer)
{
    if (!ModelState.IsValid) return BadRequest(ModelState); //400
     _db.Gamers.Add(gamer);
    await _db.SaveChangesAsync();
    //Return Created/201.
    return CreatedAtRoute("DefaultApi", new { id = gamer.Id }, gamer);    //Created/201
}
```

-->
Post: api/GamerFour
http://localhost:59537/api/GamerFour
Request Header:
Host: localhost:59537
Content-Type: application/json
Request Body:
{"Name":"Name8","Gender":"Male","Score":3000,"TeamId":1}
-->



```
POST    http://localhost:59537/api/GamerFour

Host: localhost:59537
Content-Type: application/json

Request Body
{"Name":"Name8","Gender":"Male","Score":3000,"TeamId":1}
```

-->



#	Result	Protocol	Host	URL
146	201	HTTP	localhost:59537	/api/GamerFour

```
HTTP/1.1 201 Created
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/json; charset=utf-8
Expires: -1
Location: http://localhost:59537/api/GamerFour/8
Server: Microsoft-IIS/10.0
X-AspNet-Version: 4.0.30319
X-SourceFiles: =?UTF-8?B?RDpcMV9HaXRcS0wwMVwwX0tMXDA4X1dlYkFwaV9LTFxUMDA4XE9ubGluZUdhbWVcT25saW5lR2FtZS5XZWJBcGl
X-Powered-By: ASP.NET
Date: Sat, 05 May 2018 15:26:57 GMT
Content-Length: 102

{"$id":"1","Id":8,"Name":"Name8","Gender":"Male","Score":3000,"TeamId":1,"Team":null,"GamerSkills":[]}
```

4.5.3.2. Post Request - public async Task<HttpResponseMessage> AddGamer(Gamer gamer) - Bug

```
// POST: api/Gamer4/AddGamer
[HttpPost]
[Route("AddGamer")]
public async Task<HttpResponseMessage> AddGamer(Gamer gamer)
{
    if (!ModelState.IsValid)
        return Request.CreateErrorResponse(HttpStatusCode.BadRequest,
            "ModelState is invalid");    //400

    _db.Gamers.Add(gamer);
    await _db.SaveChangesAsync();
    //Return Created/201.
    HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.Created);
    response.Headers.Location = new Uri(Request.RequestUri + "/" + gamer.Id);
    return response;    //Created/201
}
```

-->

Post: api/Gamer4/AddGamer

http://localhost:59537/api/Gamer4/AddGamer

Request Header:

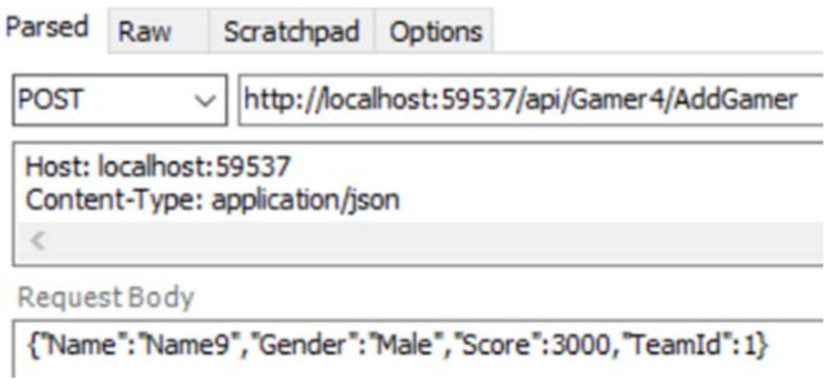
Host: localhost:59537

Content-Type: application/json

Request Body:

{"Name":"Name9","Gender":"Male","Score":3000,"TeamId":1}

-->


Parsed | Raw | Scratchpad | Options

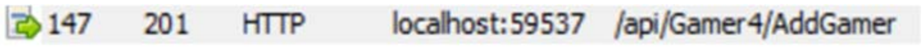
POST | http://localhost:59537/api/Gamer4/AddGamer

```
Host: localhost:59537
Content-Type: application/json
```

Request Body

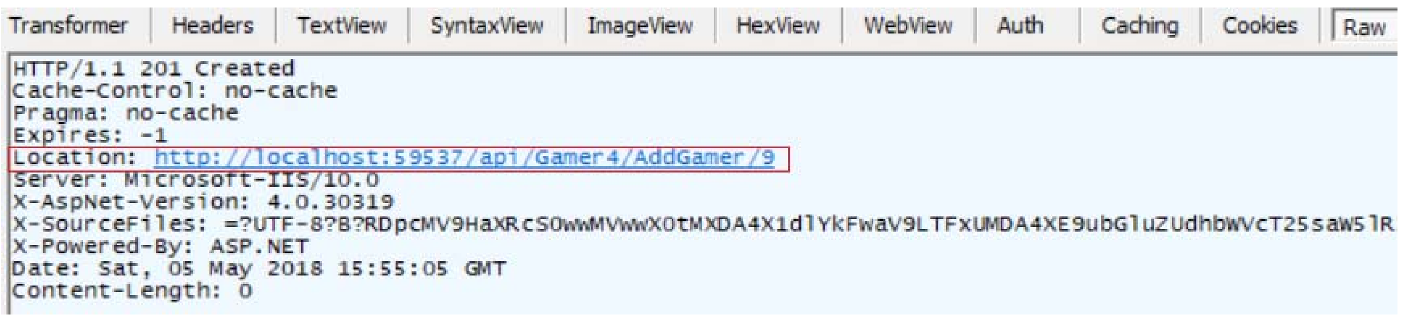
```
{"Name":"Name9","Gender":"Male","Score":3000,"TeamId":1}
```

-->



```
147    201    HTTP    localhost:59537    /api/Gamer4/AddGamer
```

-->



```
HTTP/1.1 201 Created
Cache-Control: no-cache
Pragma: no-cache
Expires: -1
Location: http://localhost:59537/api/Gamer4/AddGamer/9
Server: Microsoft-IIS/10.0
X-AspNet-Version: 4.0.30319
X-SourceFiles: =?UTF-8?B?RDpcMV9HaXRcS0wwMVVwwX0tMXDA4X1dlYkFwaV9LTFxUMDA4XE9ubGluZUdhbWVcT25saW5lR
X-Powered-By: ASP.NET
Date: Sat, 05 May 2018 15:55:05 GMT
Content-Length: 0
```

The location is totally not right.

4.5.3.3. Post Request - public async Task<IHttpActionResult> AddGamer2(Gamer gamer) - Bug

```
//IHttpActionResult is from Web API 2
// POST: api/Gamer4/AddGamer2
[Route("AddGamer2")]
[HttpPost]
[ResponseType(typeof(Gamer))]
public async Task<IHttpActionResult> AddGamer2(Gamer gamer)
{
    if (!ModelState.IsValid) return BadRequest(ModelState); //400
    _db.Gamers.Add(gamer);
    await _db.SaveChangesAsync();
    //Return Created/201.
    return CreatedAtRoute("DefaultApi", new { id = gamer.Id }, gamer);    //Created/201
}
```

-->

Post: api/Gamer4/AddGamer2

http://localhost:59537/api/Gamer4/AddGamer2

Request Header:

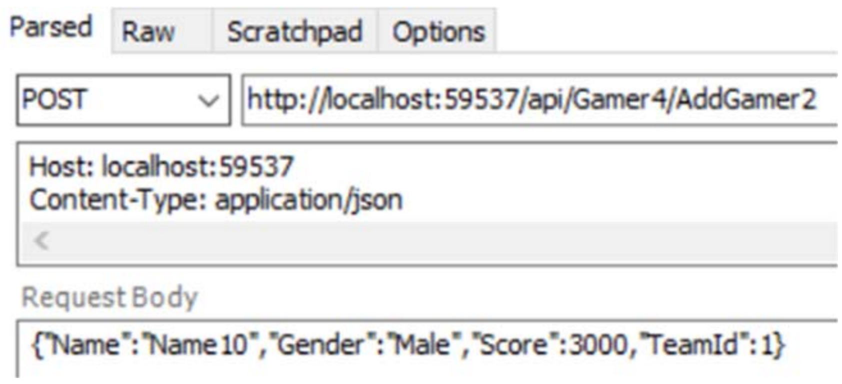
Host: localhost:59537

Content-Type: application/json

Request Body:

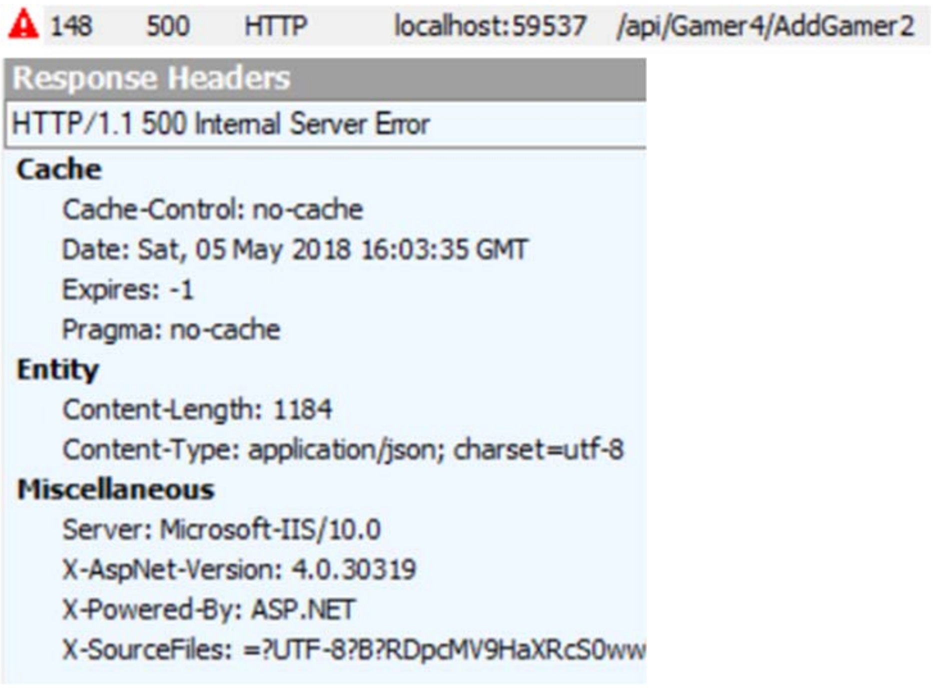
{"Name":"Name10","Gender":"Male","Score":3000,"TeamId":1}

-->



```
POST http://localhost:59537/api/Gamer4/AddGamer2
Host: localhost:59537
Content-Type: application/json

Request Body
{"Name":"Name10","Gender":"Male","Score":3000,"TeamId":1}
```

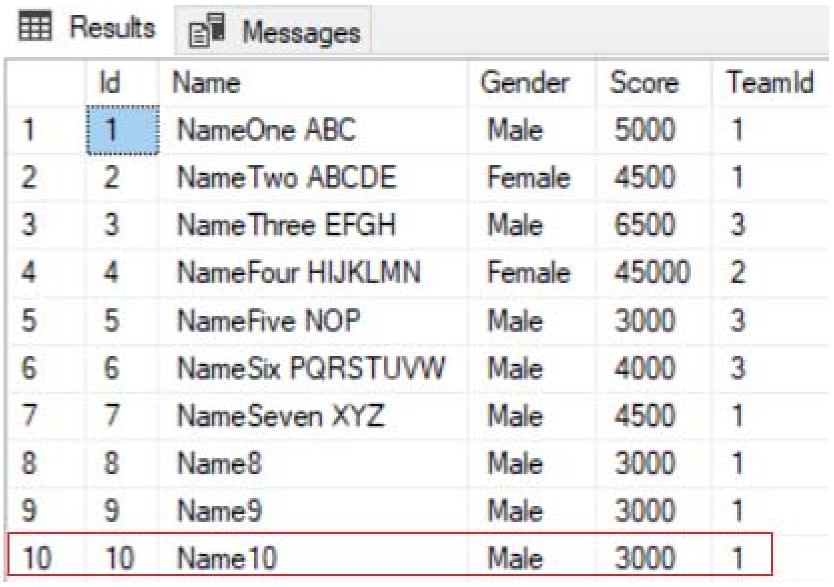
-->



```
148  500  HTTP  localhost:59537  /api/Gamer4/AddGamer2

Response Headers
HTTP/1.1 500 Internal Server Error
Cache
    Cache-Control: no-cache
    Date: Sat, 05 May 2018 16:03:35 GMT
    Expires: -1
    Pragma: no-cache
Entity
    Content-Length: 1184
    Content-Type: application/json; charset=utf-8
Miscellaneous
    Server: Microsoft-IIS/10.0
    X-AspNet-Version: 4.0.30319
    X-Powered-By: ASP.NET
    X-SourceFiles: =?UTF-8?B?RDpcMV9HaXRcS0ww
```

-->

But SQL still has the record



	Id	Name	Gender	Score	TeamId
1	1	NameOne ABC	Male	5000	1
2	2	NameTwo ABCDE	Female	4500	1
3	3	NameThree EFGH	Male	6500	3
4	4	NameFour HIJKLMN	Female	45000	2
5	5	NameFive NOP	Male	3000	3
6	6	NameSix PQRSTUVW	Male	4000	3
7	7	NameSeven XYZ	Male	4500	1
8	8	Name8	Male	3000	1
9	9	Name9	Male	3000	1
10	10	Name10	Male	3000	1

4.5.3.4. Post Request - public async Task<HttpResponseMessage> AddGamer3(Gamer gamer) - Fix Bug

```
// GET: api/Gamer4/1
[ResponseType(typeof(Gamer))]
[Route("{id:int}", Name = "GetGamerById")]
```

```
public async Task<IHttpActionResult> GetGamerById(int id)
{
    Gamer gamer = await _db.Gamers.FindAsync(id);
    if (gamer == null) return NotFound();   //404
    return Ok(gamer);
}

// POST: api/Gamer4/AddGamer3
[HttpPost]
[Route("AddGamer3")]
public async Task<HttpResponseMessage> AddGamer3(Gamer gamer)
{
    if (!ModelState.IsValid)
        return Request.CreateErrorResponse(HttpStatusCode.BadRequest,
            "ModelState is invalid");    //400
    _db.Gamers.Add(gamer);
    await _db.SaveChangesAsync();
    //Return Created/201.
    HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.Created);
    response.Headers.Location = new
                Uri(Url.Link("GetGamerById", new { id = gamer.Id }));
    return response;    //Created/201
}
```

-->

Post: api/Gamer4/AddGamer3

http://localhost:59537/api/Gamer4/AddGamer3

Request Header:

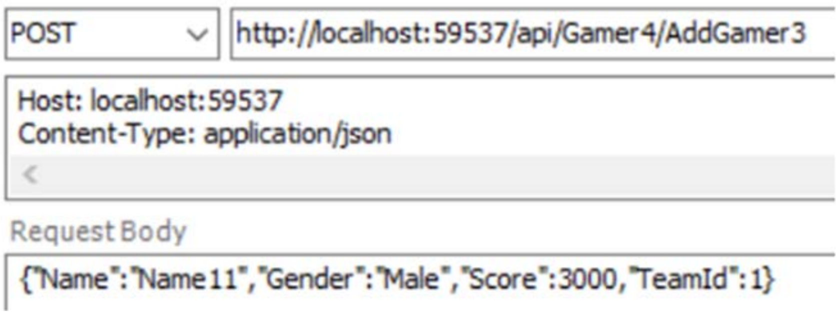
Host: localhost:59537

Content-Type: application/json

Request Body:

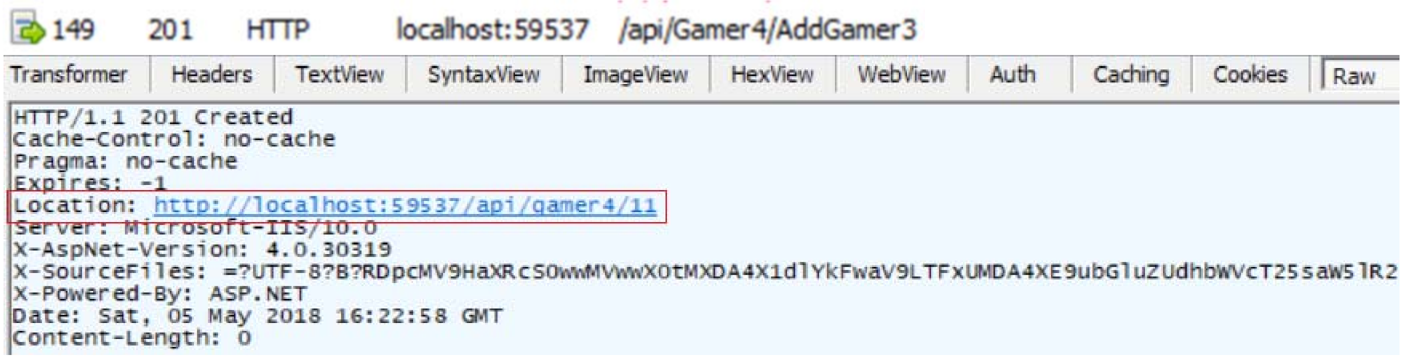
{"Name":"Name11","Gender":"Male","Score":3000,"TeamId":1}

-->



```
POST    http://localhost:59537/api/Gamer4/AddGamer3

Host: localhost:59537
Content-Type: application/json

Request Body
{"Name":"Name11","Gender":"Male","Score":3000,"TeamId":1}
```

-->

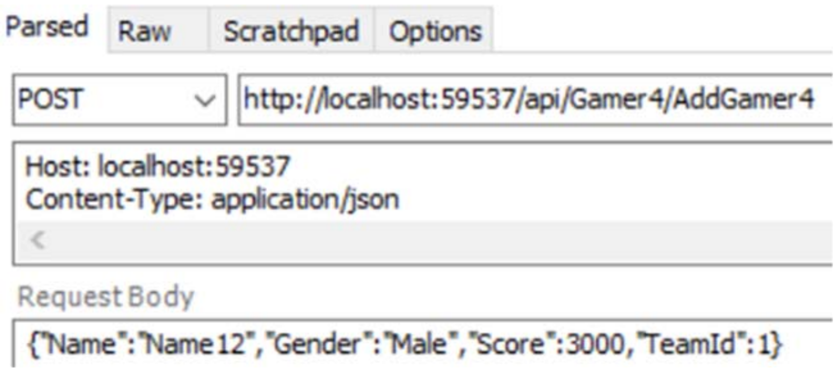


```
149    201    HTTP    localhost:59537    /api/Gamer4/AddGamer3
```

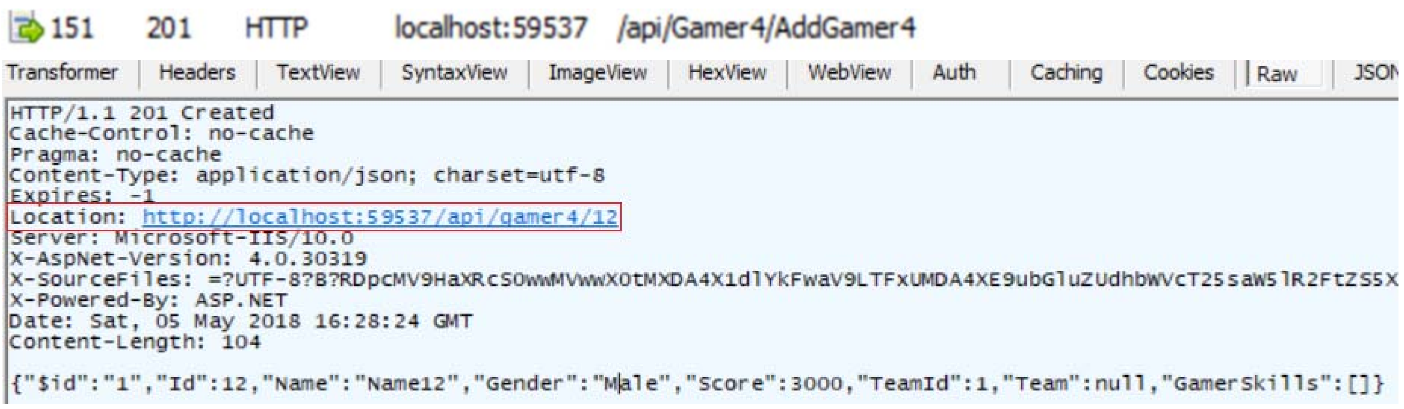
```
HTTP/1.1 201 Created
Cache-Control: no-cache
Pragma: no-cache
Expires: -1
Location: http://localhost:59537/api/gamer4/11
Server: Microsoft-IIS/10.0
X-AspNet-Version: 4.0.30319
X-SourceFiles: =?UTF-8?B?RDpcMV9HaXRcS0wwMVVwwX0tMXDA4X1dlYkFwaV9LTFxUMDA4XE9ubGluZUdhbWVcT25saW5lR2
X-Powered-By: ASP.NET
Date: Sat, 05 May 2018 16:22:58 GMT
Content-Length: 0
```

4.5.3.5. Post Request - public async Task<IHttpActionResult> AddGamer4(Gamer gamer) - Fix Bug

```
// GET: api/Gamer4/1
[ResponseType(typeof(Gamer))]
[Route("{id:int}", Name = "GetGamerById")]
public async Task<IHttpActionResult> GetGamerById(int id)
{
    Gamer gamer = await _db.Gamers.FindAsync(id);
    if (gamer == null) return NotFound();   //404
    return Ok(gamer);
}

// POST: api/Gamer4/AddGamer4
[HttpPost]
[Route("AddGamer4")]
public async Task<IHttpActionResult> AddGamer4(Gamer gamer)
{
    if (!ModelState.IsValid) return BadRequest(ModelState); //400
    _db.Gamers.Add(gamer);
    await _db.SaveChangesAsync();
    //Return Created/201.
    return CreatedAtRoute("GetGamerById", new { id = gamer.Id }, gamer);    //Created/201
}
```
-->
Post: api/Gamer4/AddGamer4
http://localhost:59537/api/Gamer4/AddGamer4
Request Header:
Host: localhost:59537
Content-Type: application/json
Request Body:
{"Name":"Name12","Gender":"Male","Score":3000,"TeamId":1}
-->



```
POST http://localhost:59537/api/Gamer4/AddGamer4
Host: localhost:59537
Content-Type: application/json

{"Name":"Name12","Gender":"Male","Score":3000,"TeamId":1}
```
-->

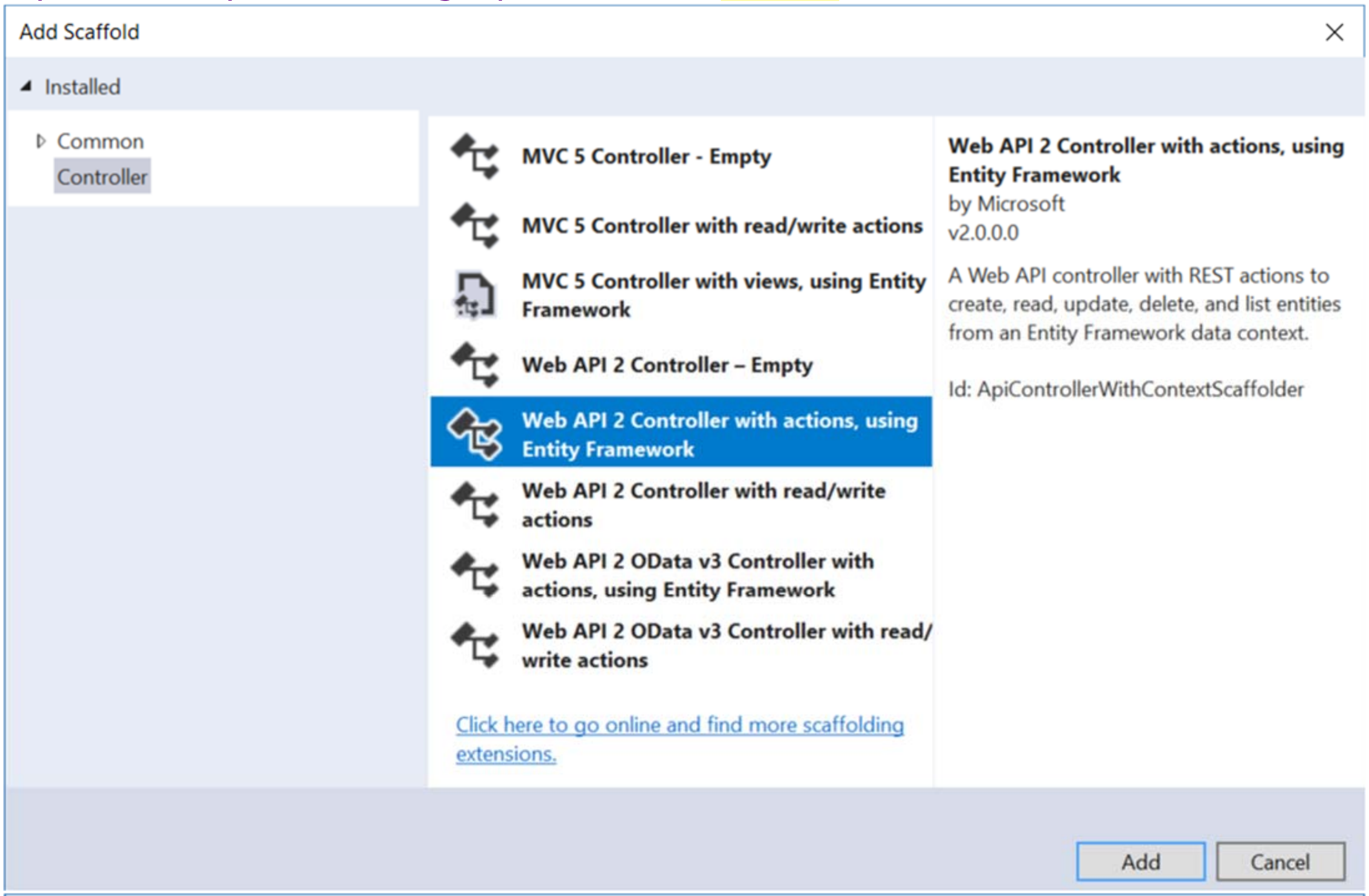


```
HTTP/1.1 201 Created
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/json; charset=utf-8
Expires: -1
Location: http://localhost:59537/api/gamer4/12
Server: Microsoft-IIS/10.0
X-AspNet-Version: 4.0.30319
X-SourceFiles: =?UTF-8?B?RDpcMV9HaXRcS0wwMVVwwX0tMXDA4X1dlYkFwaV9LTFxUMDA4XE9ubGluZUdhbWVcT25saW5lR2FtZS5X
X-Powered-By: ASP.NET
Date: Sat, 05 May 2018 16:28:24 GMT
Content-Length: 104

{"$id":"1","Id":12,"Name":"Name12","Gender":"Male","Score":3000,"TeamId":1,"Team":null,"GamerSkills":[]}
```

4.6. OnlineGame.WebApi/Controllers/Api/GamerFiveController.cs - Route names

4.6.1. OnlineGame.WebApi/Controllers/Api/GamerFiveController.cs

Controllers/Api folder --> Right Click --> Add --> Controller
--> **Web API 2 Controller with actions, using Entity Framework**
--> **GamerFiveController**
if you have any error message, please ensure re-build whole solutions.



Add Scaffold

Installed
- Common
  - Controller

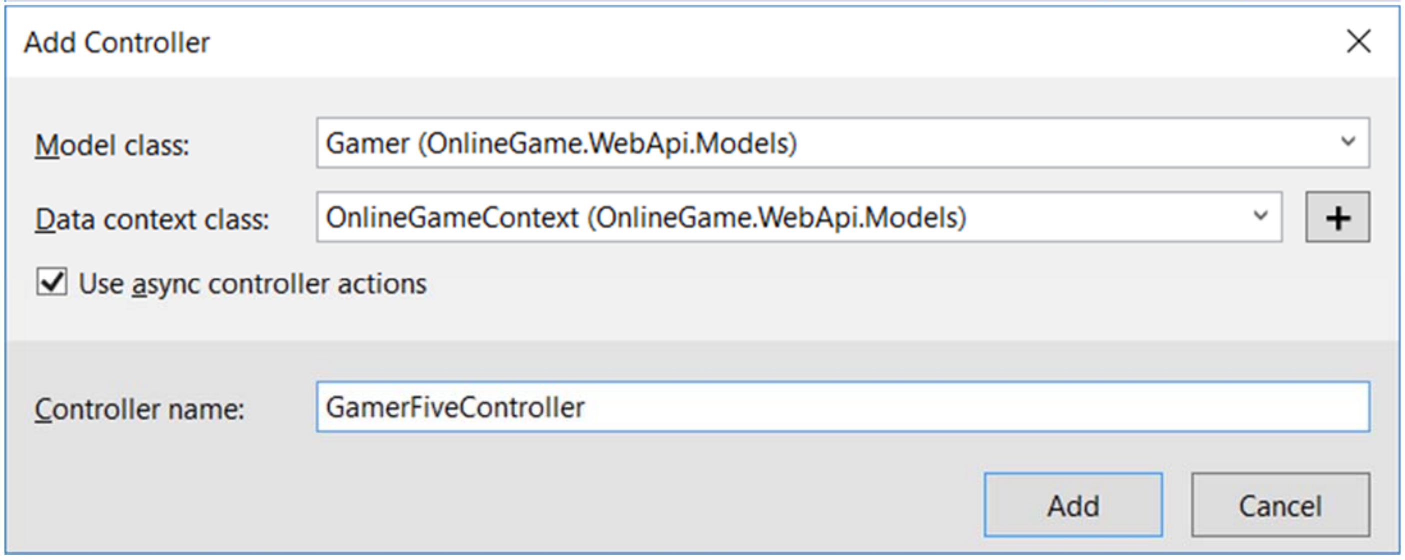
- MVC 5 Controller - Empty
- MVC 5 Controller with read/write actions
- MVC 5 Controller with views, using Entity Framework
- Web API 2 Controller – Empty
- Web API 2 Controller with actions, using Entity Framework
- Web API 2 Controller with read/write actions
- Web API 2 OData v3 Controller with actions, using Entity Framework
- Web API 2 OData v3 Controller with read/write actions

Click here to go online and find more scaffolding extensions.

Web API 2 Controller with actions, using Entity Framework
by Microsoft
v2.0.0.0

A Web API controller with REST actions to create, read, update, delete, and list entities from an Entity Framework data context.

Id: ApiControllerWithContextScaffolder



Add Controller

Model class: Gamer (OnlineGame.WebApi.Models)

Data context class: OnlineGameContext (OnlineGame.WebApi.Models)

☑ Use async controller actions

Controller name: GamerFiveController

4.6.2. OnlineGame.WebApi/Controllers/Api/GamerFiveController.cs

```
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Threading.Tasks;
using System.Web.Http;
using System.Web.Http.Description;
using OnlineGame.WebApi.Models;

namespace OnlineGame.WebApi.Controllers.Api
{
    [RoutePrefix("api/gamer5")]
    public class GamerFiveController : ApiController
    {
        private OnlineGameContext _db = new OnlineGameContext();

        // GET: api/gamer5
        [Route("")]
        public IQueryable<Gamer> GetGamers()
        {
            return _db.Gamers;
        }

        // GET: api/gamer5/GetGamers2
        [Route("GetGamers2")]
        public IHttpActionResult GetGamers2()
        {
            return Ok(_db.Gamers);
        }

        // GET: api/gamer5/GetGamers3
        [Route("GetGamers3")]
        public HttpResponseMessage GetGamers3()
        {
            return Request.CreateResponse(_db.Gamers);
        }

        // GET: api/gamer5/GetGamer/1
        [Route("GetGamer/{id:int}")]
        [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamer(int id)
        {
            Gamer gamer = await _db.Gamers.FindAsync(id);
            if (gamer == null) return NotFound();   //404
            return Ok(gamer);
        }

        // GET: api/gamer5/GetGamer2/1
        [Route("GetGamer2/{id:int}")]
        [ResponseType(typeof(Gamer))]
        public async Task<HttpResponseMessage> GetGamer2(int id)
        {
            Gamer gamer = await _db.Gamers.FindAsync(id);
            if (gamer == null)
                return Request.CreateErrorResponse(HttpStatusCode.NotFound,
                "Gamer not found"); //404
            return Request.CreateResponse(gamer);
        }

        // GET: api/gamer5/GetGamer3/1
        [Route("GetGamer3/{id:int}")]
```

```
         [ResponseType(typeof(Gamer))]
        public async Task<IHttpActionResult> GetGamer3(int id)
        {
             Gamer gamer = await _db.Gamers.FindAsync(id);
            if (gamer == null)
                return Content(HttpStatusCode.NotFound, "Gamer not found"); //404
            return Ok(gamer);
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing) _db.Dispose();
            base.Dispose(disposing);
        }

        private bool GamerExists(int id)
        {
            return _db.Gamers.Count(e => e.Id == id) > 0;
        }
    }
}

/*
10.
IHttpActionResult vs HttpResponseMessage
10.1.
IHttpActionResult
10.1.1.
HttpResponseMessage is from Web API 1
IHttpActionResult is from Web API 2
10.1.2.
IHttpActionResult make code cleaner.
10.1.3.
The following type implements IHttpActionResult interface.
Unauthorized()
BadRequest()
NotFound()
Created()
OK()
InternalServerError()
*/
```