(T9)Api 的 Version
CourseGUID 4c5822ff-7111-4e25-a336-ef18d48d54bd
==========
(T9)Api 的 Version
(T9-1)前置設定
(T9-2)討論用 URI 設定 Api 的 Version
(T9-3)討論用 URI、RoutePrefix 屬性、Route 屬性設定 Api 的 Version
(T9-4)討論用 QueryString 設定 Api 的 Version
(T9-5)討論用 RequestCustomHeaderProperty 設定 Api 的 Version
(T9-6)討論用 RequestHeaderAcceptProperty 設定 Api 的 Version
(T9-7)討論用 CustomMediaTypes 設定 Api 的 Version
==========
0. What to Learn
---
1. Why is versioning required in Web API
---
2. OnlineGame Solution
2.1. OnlineGame Solution
2.2. OnlineGame.Data
2.3. OnlineGame.WebApiA
2.4. OnlineGame.WebApiB
2.5. OnlineGame.WebApiC
2.6. OnlineGame.WebApiD
2.7. OnlineGame.WebApiE
2.8. OnlineGame.WebApiF
---
3. OnlineGame.Data
3.1. OnlineGame.Data/GamerV1
3.2. OnlineGame.Data/GamerV2
---
4. OnlineGame.WebApiA - Version by URI
4.1. What to do - Version by URI
4.2. OnlineGame.WebApiA/Controllers/GamerV1Controller.cs - Version by URI
4.3. OnlineGame.WebApiA/Controllers/GamerV2Controller.cs - Version by URI
4.4. OnlineGame.WebApiA/App_Start/WebApiConfig.cs - Version by URI
---
5. OnlineGame.WebApiB - Version by URI with RoutePrefix and Route Attribute
5.1. What to do - Version by URI with RoutePrefix and Route Attribute
5.2. OnlineGame.WebApiB/Controllers/GamerV1Controller.cs - Version by URI with RoutePrefix and Route Attribute
5.3. OnlineGame.WebApiB/Controllers/GamerV2Controller.cs - Version by URI with RoutePrefix and Route Attribute
---
6. OnlineGame.WebApiC - Version by querystring
6.1. What to do - Version by querystring
6.2. OnlineGame.WebApiC/WebApiShare/CustomControllerSelector.cs - Version by querystring
6.3. OnlineGame.WebApiC/App_Start/WebApiConfig.cs - Version by querystring
6.4. OnlineGame.WebApiC/Controllers/GamerV1Controller.cs - Version by querystring
6.5. OnlineGame.WebApiC/Controllers/GamerV2Controller.cs - Version by querystring
---
7. OnlineGame.WebApiD - Version by the request custom header property
7.1. What to do - Version by the request custom header property
7.2. OnlineGame.WebApiD/WebApiShare/CustomControllerSelector.cs - Version by the request custom header property

7.3. OnlineGame.WebApiD/App_Start/WebApiConfig.cs - Version by the request custom header property
7.4. OnlineGame.WebApiD/Controllers/GamerV1Controller.cs - Version by the request custom header property
7.5. OnlineGame.WebApiD/Controllers/GamerV2Controller.cs - Version by the request custom header property
---
8. OnlineGame.WebApiE - Version by the request header Accept property
8.1. What to do
8.2. OnlineGame.WebApiE/WebApiShare/CustomControllerSelector.cs - Version by the request header Accept property
8.3. OnlineGame.WebApiE/App_Start/WebApiConfig.cs - Version by the request header Accept property
8.4. OnlineGame.WebApiE/Controllers/GamerV1Controller.cs - Version by the request header Accept property
8.5. OnlineGame.WebApiE/Controllers/GamerV2Controller.cs - Version by the request header Accept property
---
9. OnlineGame.WebApiF - version by custom media types
9.1. What to do - version by custom media types
9.2. OnlineGame.WebApiF/WebApiShare/CustomControllerSelector.cs - version by custom media types
9.3. OnlineGame.WebApiF/App_Start/WebApiConfig.cs - version by custom media types
9.4. OnlineGame.WebApiF/Controllers/GamerV1Controller.cs - version by custom media types
9.5. OnlineGame.WebApiF/Controllers/GamerV2Controller.cs - version by custom media types
==========

0. What to Learn

The tutorial will discuss ...
Version by URI
Version by URI with RoutePrefix and Route Attribute
Version by querystring
Version by the request custom header property
Version by the request header Accept property
version by custom media types

本堂課討論
關於 Version by URI
關於 Version by URI with RoutePrefix and Route Attribute
關於 Version by querystring
關於 Version by the request custom header property
關於 Version by the request header Accept property
關於 version by custom media types

1. Why is versioning required in Web API

1.
We publish our version 1 API, and our users started to use our version 1 api.
2.
After a few months, we update our API and create version 2 API.
We can not just delete version 1 API, because there are users still use it.

Thus, we have to keep version 1 and version together in our API.
3.
We may tell our user the expire day of version 1 API and encourage users to start to use version 2.

2. OnlineGame Solution

2.1. OnlineGame Solution

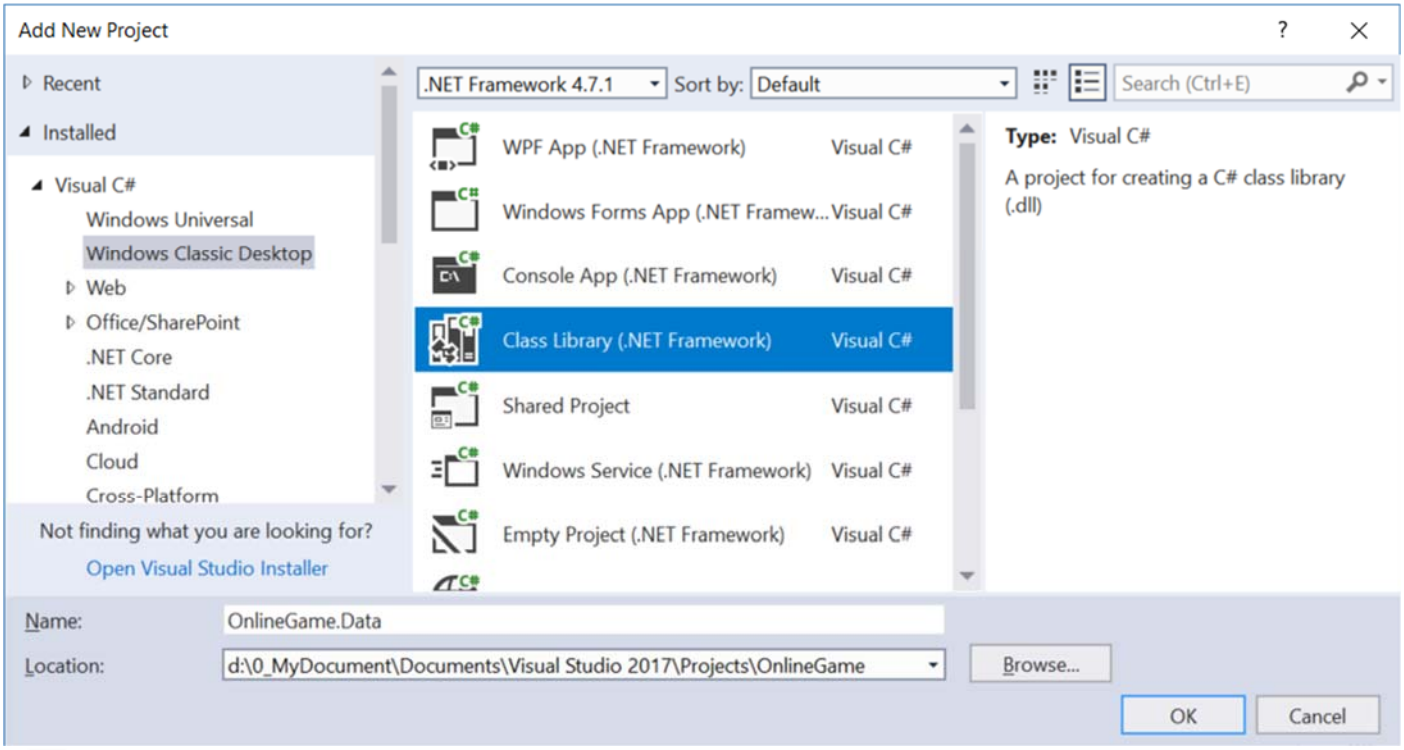
File --> New --> Project... -->

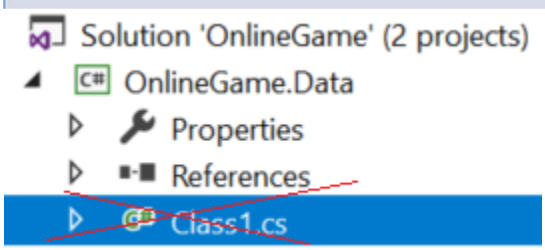
Other Project Types --> Visual Studio Solutions --> Blank Solution
-->
Name: **OnlineGame**

2.2. OnlineGame.Data

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

Visual C# --> **Class Library (.NET Framework)**
-->
Name:
OnlineGame.Data
-->
Delete Class1.cs



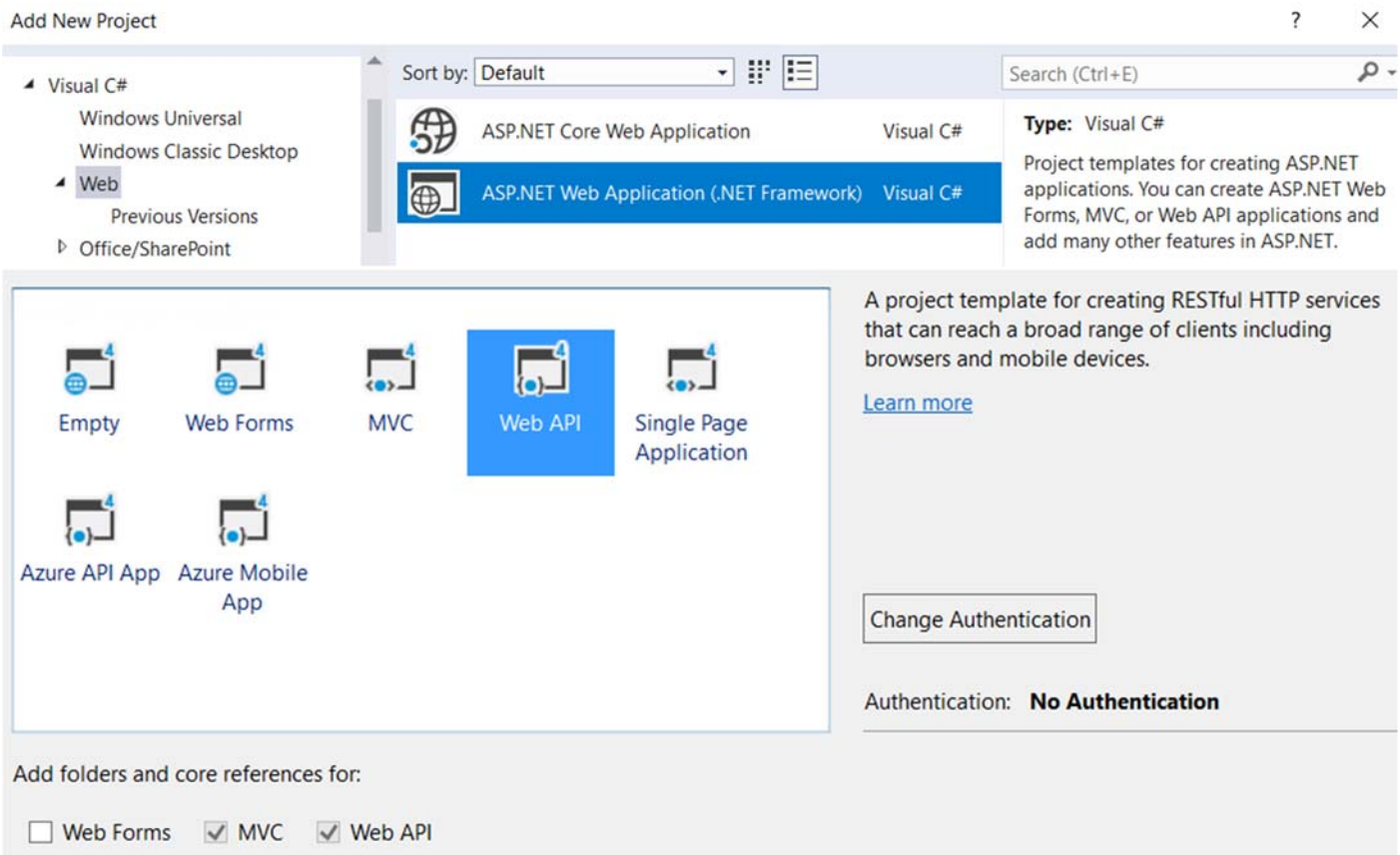


2.3. OnlineGame.WebApiA

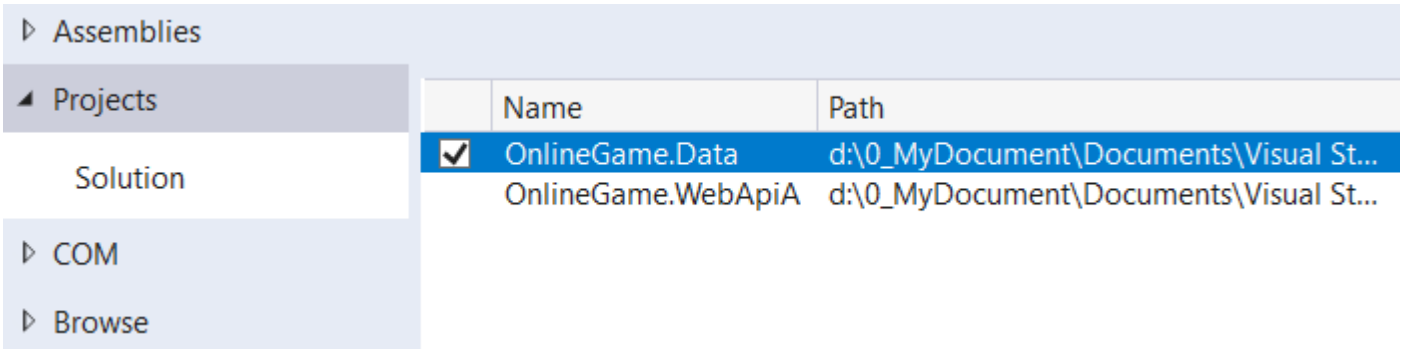
Solutions Name --> Add --> New Project -->

Visual C# --> Web --> ASP.NETWeb Application (.Net Framework)
-->

Name: **OnlineGame.WebApiA**
--> Select "**Web API**"
--> OK



Add Reference:
OnlineGame.Data

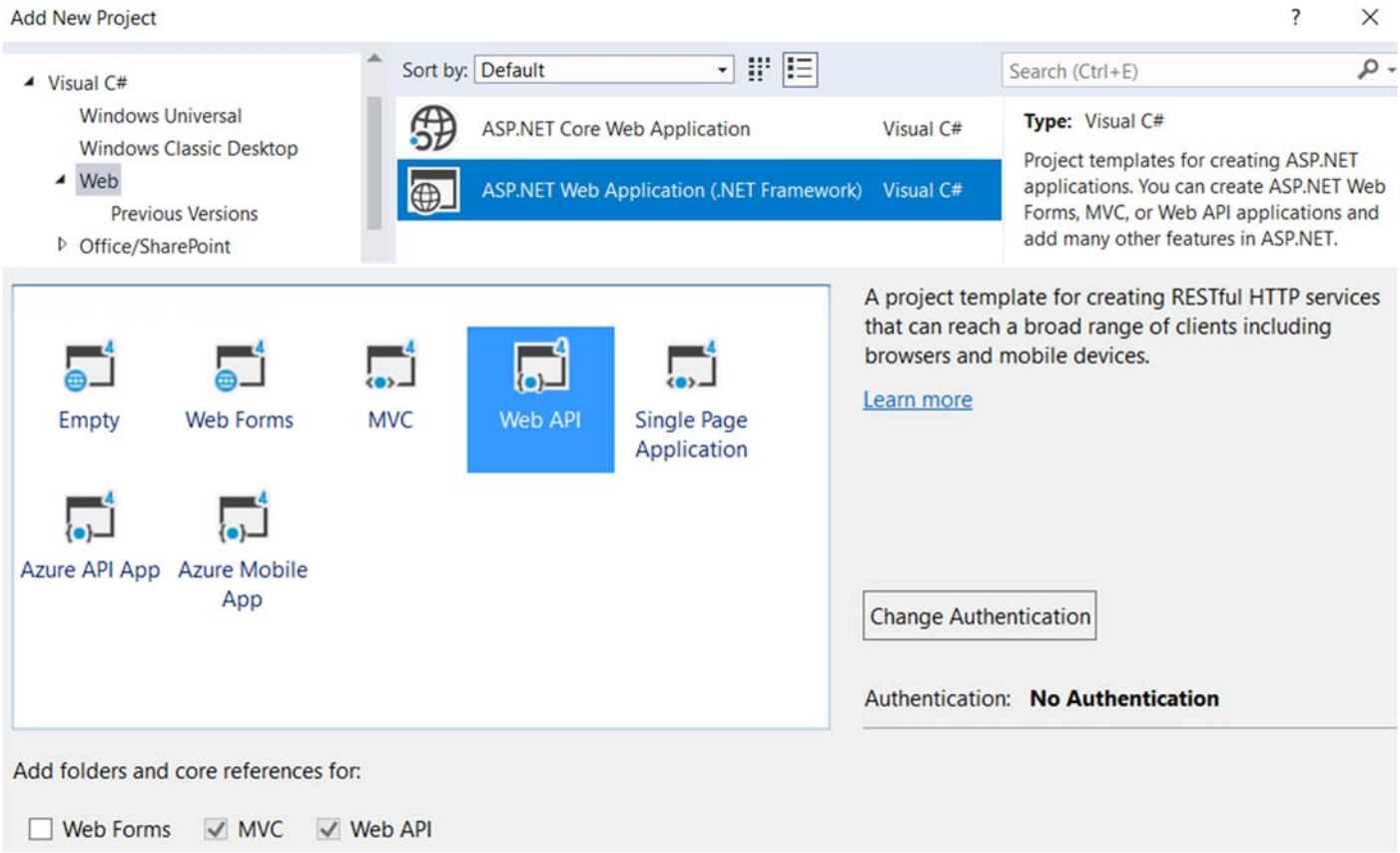


2.4. OnlineGame.WebApiB

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

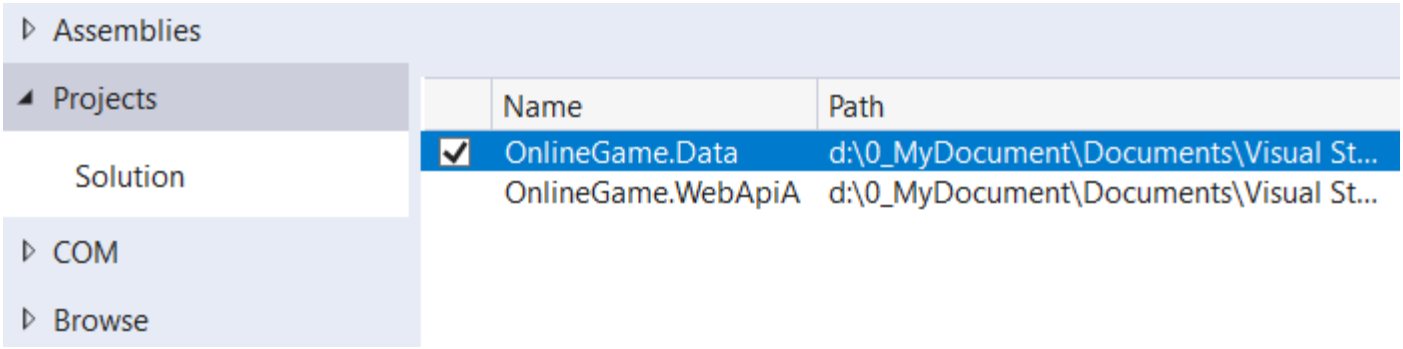
Visual C# --> Web --> ASP.NETWeb Application (.Net Framework)
-->

Name: **OnlineGame.WebApiB**
--> Select "**Web API**"
--> OK



Add Reference:

OnlineGame.Data



2.5. OnlineGame.WebApiC

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

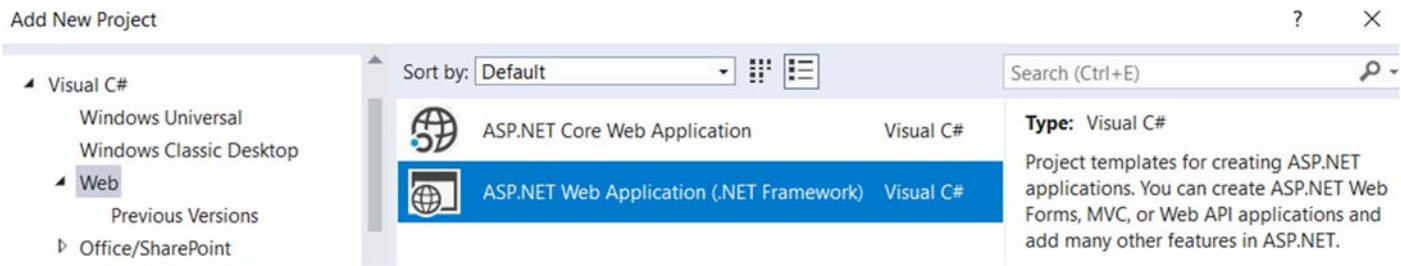
Visual C# --> Web --> ASP.NETWeb Application (.Net Framework)

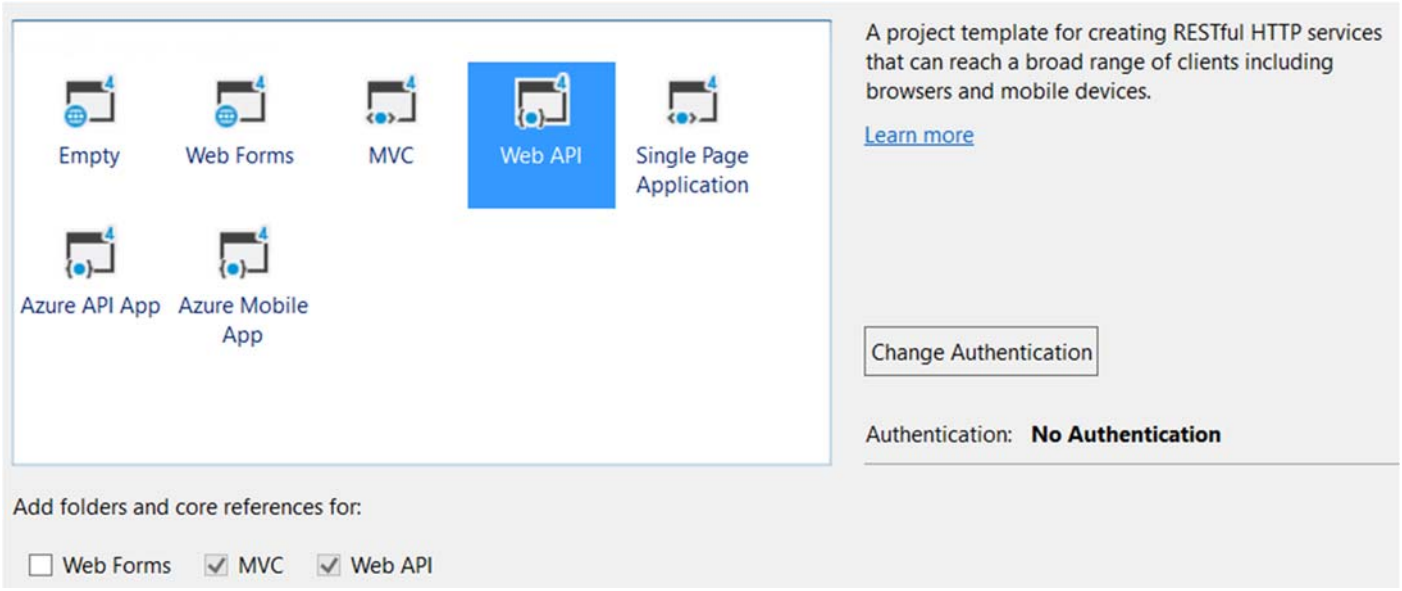
-->

Name: **OnlineGame.WebApiC**

--> Select "**Web API**"

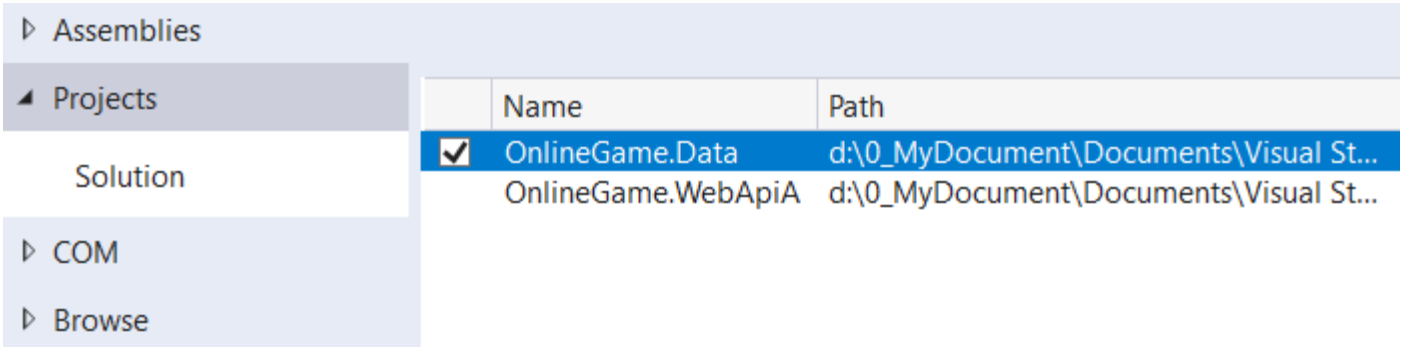
--> OK





Add Reference:

OnlineGame.Data



2.6. OnlineGame.WebApiD

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

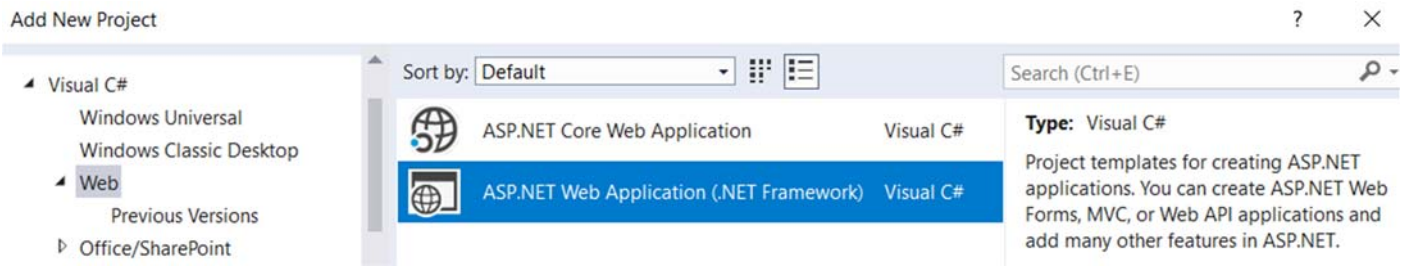
Visual C# --> Web --> ASP.NETWeb Application (.Net Framework)

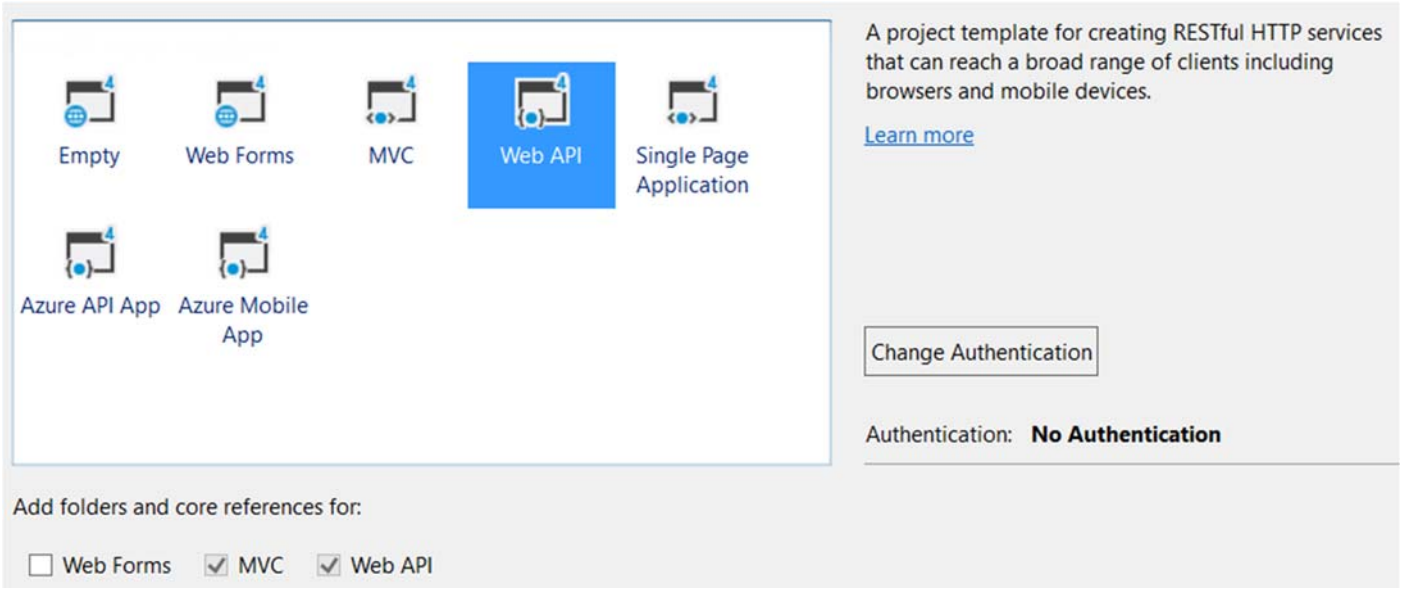
-->

Name: **OnlineGame.WebApiD**

--> Select "**Web API**"

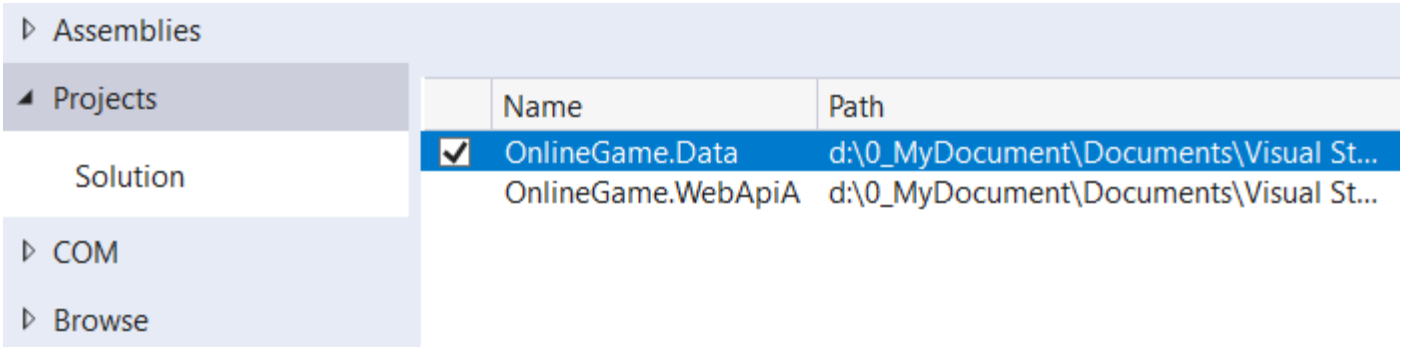
--> OK





Add Reference:

OnlineGame.Data



2.7. OnlineGame.WebApiE

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

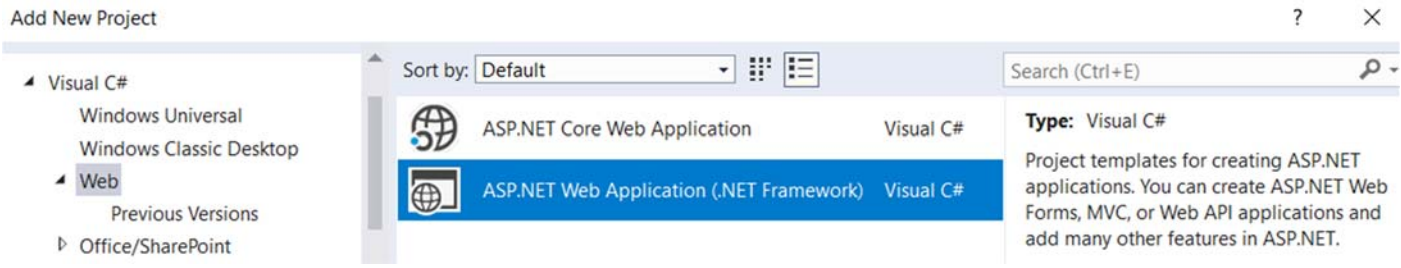
Visual C# --> Web --> ASP.NETWeb Application (.Net Framework)

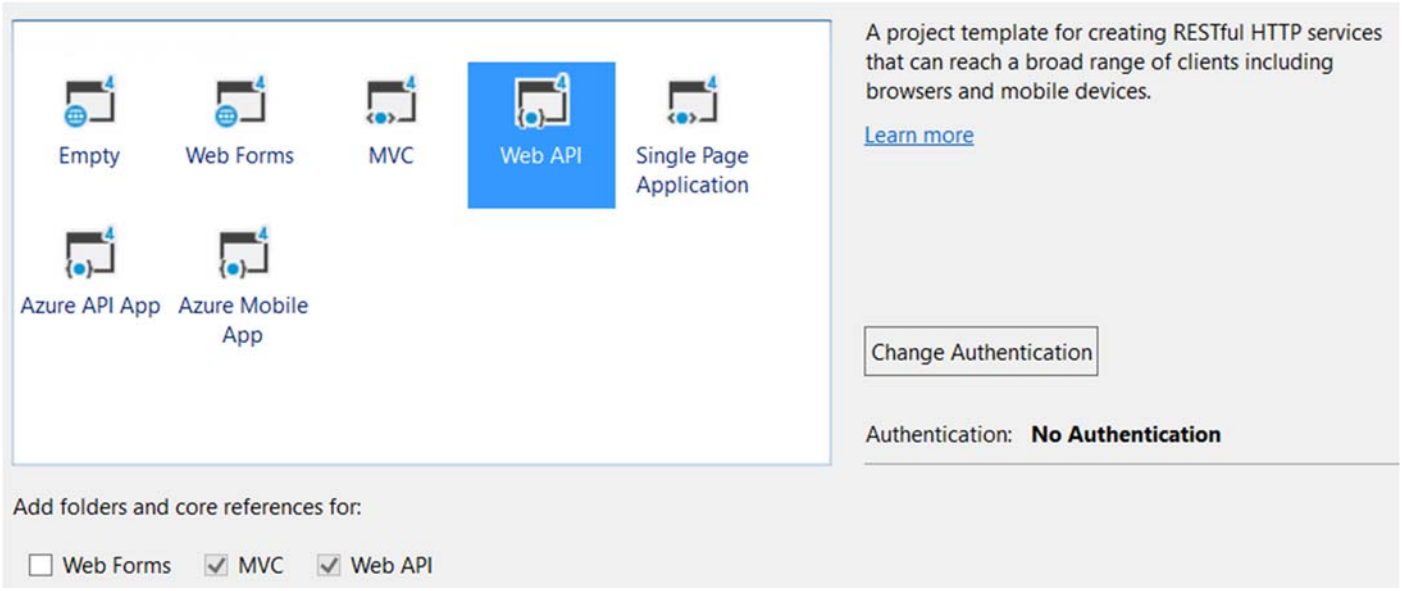
-->

Name: **OnlineGame.WebApiE**

--> Select "**Web API**"

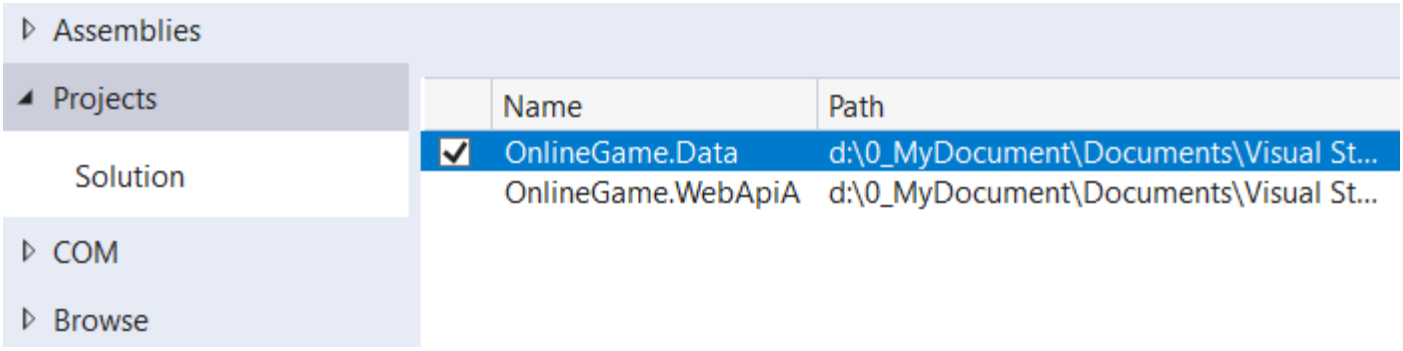
--> OK





Add Reference:

OnlineGame.Data



2.8. OnlineGame.WebApiF

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

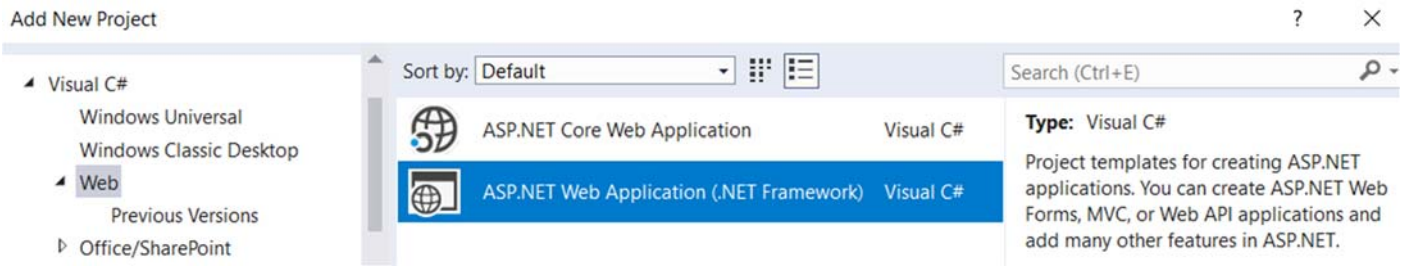
Visual C# --> Web --> ASP.NETWeb Application (.Net Framework)

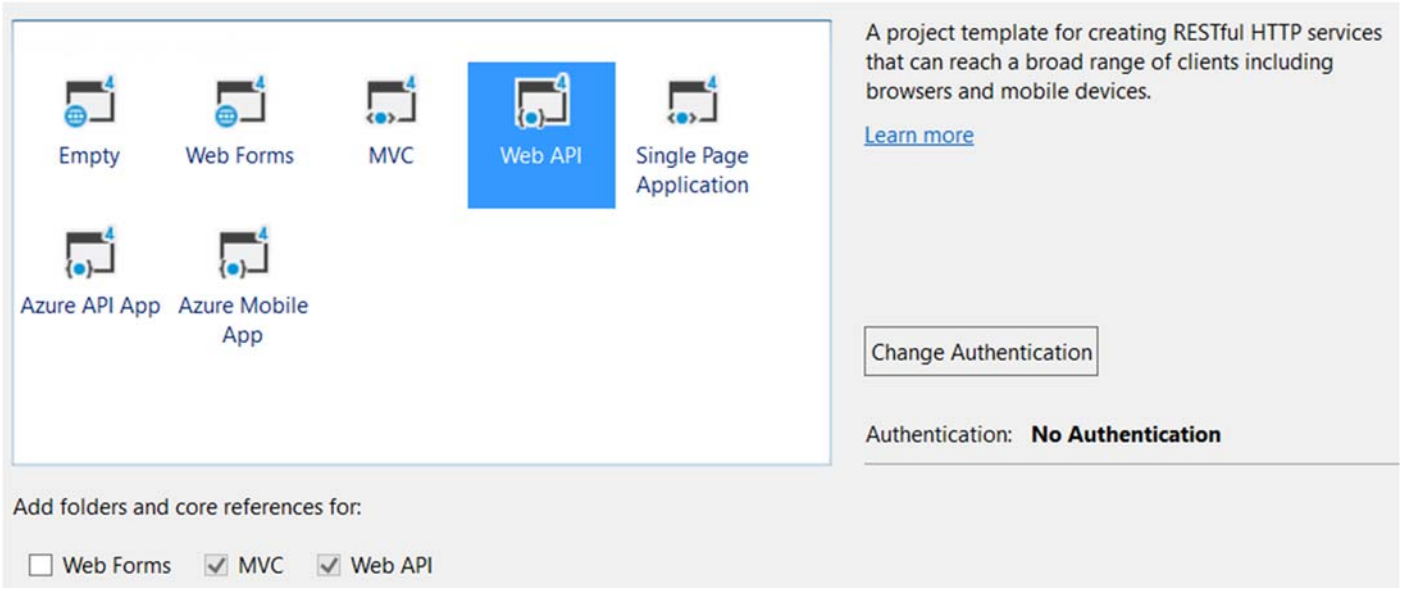
-->

Name: **OnlineGame.WebApiF**

--> Select "**Web API**"

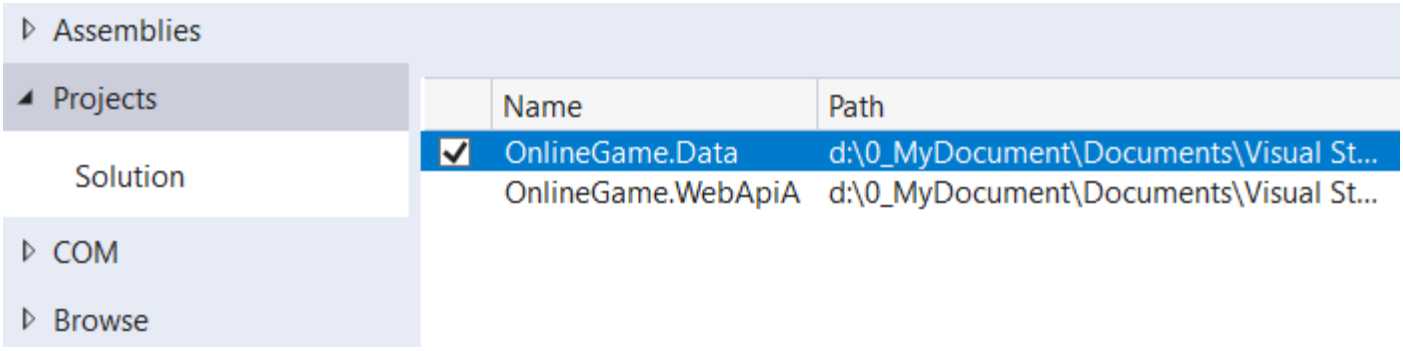
--> OK





Add Reference:

OnlineGame.Data



3. OnlineGame.Data

3.1. OnlineGame.Data/GamerV1

```
namespace OnlineGame.Data
{
    public class GamerV1
    {
        public int Id { get; set; }
        public string Name { get; set; }
    }
}
```

3.2. OnlineGame.Data/GamerV2

```
namespace OnlineGame.Data
{
    public class GamerV2
    {
        public int Id { get; set; }
        public string FirstName { get; set; }
        public string LastName { get; set; }
    }
}
```

4. OnlineGame.WebApiA - Version by URI

4.1. What to do - Version by URI

api/v1/Gamers
Call GamerV1Controller Get() action. List all gamers (version 1)
api/v1/Gamers/1
Call GamerV1Controller Get(int id) action. Return id 1 gamer(version 1)

api/v2/Gamers
Call GamerV2Controller Get() action. List all gamers (version 2)
api/v2/Gamers/1
Call GamerV2Controller Get(int id) action. Return id 1 gamer(version 2)

4.2. OnlineGame.WebApiA/Controllers/GamerV1Controller.cs - Version by URI

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;

namespace OnlineGame.WebApiA.Controllers
{
    public class GamerV1Controller : ApiController
    {
        List<GamerV1> _gamers = new List<GamerV1>
        {
            new GamerV1 { Id = 1, Name = "NameOne"},
            new GamerV1 { Id = 2, Name = "NameTwo"},
            new GamerV1{ Id = 3, Name = "NameThree"},
        };

        public IEnumerable<GamerV1> Get()
        {
            return _gamers;
        }

        public GamerV1 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

4.3. OnlineGame.WebApiA/Controllers/GamerV2Controller.cs - Version by URI

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
```

```
using OnlineGame.Data;

namespace OnlineGame.WebApiA.Controllers
{
    public class GamerV2Controller : ApiController
    {
        List<GamerV2> _gamers = new List<GamerV2>
        {
            new GamerV2 { Id = 1, FirstName = "NameFirstOne", LastName = "NameLastOne"},
            new GamerV2 { Id = 2, FirstName = "NameFirstTwo", LastName = "NameLastTwo"},
            new GamerV2 { Id = 3, FirstName = "NameFirstThree", LastName = "NameLastThree"}
        };

        public IEnumerable<GamerV2> Get()
        {
            return _gamers;
        }

        public GamerV2 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

4.4. OnlineGame.WebApiA/App_Start/WebApiConfig.cs - Version by URI

```
using System.Web.Http;
namespace OnlineGame.WebApiA
{
    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 }
            //);

            config.Routes.MapHttpRoute(
                name: "Version1",
                routeTemplate: "api/v1/Gamers/{id}",
                defaults: new { id = RouteParameter.Optional, controller = "GamerV1" }
            );

            config.Routes.MapHttpRoute(
                name: "Version2",
                routeTemplate: "api/v2/Gamers/{id}",
                defaults: new { id = RouteParameter.Optional, controller = "GamerV2" }
```

```
            );
        }
    }
}
```

5. OnlineGame.WebApiB - Version by URI with RoutePrefix and Route Attribute

5.1. What to do - Version by URI with RoutePrefix and Route Attribute

api/v1/Gamers
Call GamerV1Controller Get() action. List all gamers (version 1)
api/v1/Gamers/1
Call GamerV1Controller Get(int id) action. Return id 1 gamer(version 1)

api/v2/Gamers
Call GamerV2Controller Get() action. List all gamers (version 2)
api/v2/Gamers/1
Call GamerV2Controller Get(int id) action. Return id 1 gamer(version 2)

5.2. OnlineGame.WebApiB/Controllers/GamerV1Controller.cs - Version by URI with RoutePrefix and Route Attribute

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;
namespace OnlineGame.WebApiB.Controllers
{
    [RoutePrefix("api/v1/gamers")]
    public class GamerV1Controller : ApiController
    {
        List<GamerV1> _gamers = new List<GamerV1>
        {
            new GamerV1 { Id = 1, Name = "NameOne"},
            new GamerV1 { Id = 2, Name = "NameTwo"},
            new GamerV1{ Id = 3, Name = "NameThree"},
        };

        // GET: api/v1/gamers
        [Route("")]
        public IEnumerable<GamerV1> Get()
        {
            return _gamers;
        }

        // GET: api/v1/gamers/1
        [Route("{id}")]
        public GamerV1 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
```

```
    }
}
```

5.3. OnlineGame.WebApiB/Controllers/GamerV2Controller.cs - Version by URI with RoutePrefix and Route Attribute

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;
namespace OnlineGame.WebApiB.Controllers
{
    [RoutePrefix("api/v2/gamers")]
    public class GamerV2Controller : ApiController
    {
        List<GamerV2> _gamers = new List<GamerV2>
        {
            new GamerV2 { Id = 1, FirstName = "NameFirstOne", LastName = "NameLastOne"},
            new GamerV2 { Id = 2, FirstName = "NameFirstTwo", LastName = "NameLastTwo"},
            new GamerV2 { Id = 3, FirstName = "NameFirstThree", LastName = "NameLastThree"}
        };

        // GET: api/v2/gamers
        [Route("")]
        public IEnumerable<GamerV2> Get()
        {
            return _gamers;
        }

        // GET: api/v2/gamers/1
        [Route("{id}")]
        public GamerV2 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

6. OnlineGame.WebApiC - Version by querystring

6.1. What to do - Version by querystring

Reference:
http://csharp-video-tutorials.blogspot.com.au/2017/02/web-api-versioning-using-querystring.html

api/Gamer**?v=1**
Call GamerV1Controller Get() action. List all gamers (version 1)
api/Gamer/1**?v=1**
Call GamerV1Controller Get(int id) action. Return id 1 gamer(version 1)

api/Gamer**?v=2**
Call GamerV2Controller Get() action. List all gamers (version 2)
api/Gamer/1**?v=2**

Call GamerV2Controller Get(int id) action. Return id 1 gamer(version 2)
-->
In Web API, **DefaultHttpControllerSelector** class **SelectController()** method selects the controller and action based on URI.
E.g.
/api/gamer/1
By default, **SelectController()** will takes "**gamer**" as the controller name and call "**GamerController**".
However, it does not fit our requirement that we want query string "**?v=1**" call "**GamerV1Controller**" and query string "**?v=2**" call "**GamerV2Controller**"
Therefore, we need a "**CustomControllerSelector**"

6.2. OnlineGame.WebApiC/WebApiShare/CustomControllerSelector.cs - Version by querystring

```
using System.Collections.Generic;
using System.Collections.Specialized;
using  System.Net.Http;
using System.Web;
using System.Web.Http;
using System.Web.Http.Controllers;
using System.Web.Http.Dispatcher;
using System.Web.Http.Routing;

namespace OnlineGame.WebApiC.WebApiShare
{
    public class CustomControllerSelector : DefaultHttpControllerSelector
    {
        private HttpConfiguration _configuration;

        public CustomControllerSelector(HttpConfiguration configuration) : base(configuration)
        {
            _configuration = configuration;
        }

        public override HttpControllerDescriptor SelectController(HttpRequestMessage request)
        {
            //1.
            //Get all API controllers
            // GetControllerMapping returns all controllers which extend ApiController
            IDictionary<string, HttpControllerDescriptor> controllers =
                GetControllerMapping();
            //request.GetRouteData() returns controller name and parameter values from the request URI
            IHttpRouteData routeData = request.GetRouteData();

            //2.
            //Get Controller Name
            // routeData.Values["controller"].ToString() returns
            // the controller name from route data.
            // In this case, the controller name is "Gamers".
            string controllerName =
                routeData.Values["controller"].ToString();

            //3.
            //Set default versionNumber
```

```
            // Default version number to 1
            string versionNumber = "1";

            //4.
            //Get QueryString value
            NameValueCollection queryString =
                HttpUtility.ParseQueryString(request.RequestUri.Query);
            if (queryString["v"] != null) versionNumber = queryString["v"];

            //5.
            //Get the versionNumber from query string.
            // if versionNumber==1, then controllerName=controllerName+"V1"
            // if versionNumber==2, then controllerName=controllerName+"V2"
            controllerName =
                controllerName +
                (versionNumber == "1" ? "V1" : "V2");

            //6.
            //Find the Controller by the name
            HttpControllerDescriptor controllerDescriptor;
            if (controllers.TryGetValue(controllerName, out controllerDescriptor))
                return controllerDescriptor;
            return null;
        }
    }
}
```

6.3. OnlineGame.WebApiC/App_Start/WebApiConfig.cs - Version by querystring

```
using System.Web.Http;
using System.Web.Http.Dispatcher;
using OnlineGame.WebApiC.WebApiShare;

namespace OnlineGame.WebApiC
{
    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 }
            );

            //Replace the default controller selector,IHttpControllerSelector,
            //with our custom controller selector,CustomControllerSelector.
            config.Services.Replace(typeof(IHttpControllerSelector),
                new CustomControllerSelector(config));
            }
```

```
    }
}
```

6.4. OnlineGame.WebApiC/Controllers/GamerV1Controller.cs - Version by querystring

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;

namespace OnlineGame.WebApiC.Controllers
{
    public class GamerV1Controller : ApiController
    {
        List<GamerV1> _gamers = new List<GamerV1>
        {
            new GamerV1 { Id = 1, Name = "NameOne"},
            new GamerV1 { Id = 2, Name = "NameTwo"},
            new GamerV1{ Id = 3, Name = "NameThree"},
        };

        public IEnumerable<GamerV1> Get()
        {
            return _gamers;
        }

        public GamerV1 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

6.5. OnlineGame.WebApiC/Controllers/GamerV2Controller.cs - Version by querystring

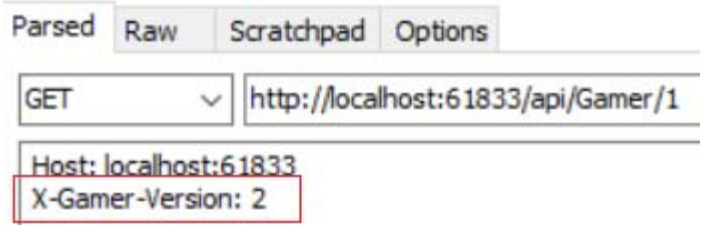
```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;
namespace OnlineGame.WebApiC.Controllers
{
    public class GamerV2Controller : ApiController
    {
        List<GamerV2> _gamers = new List<GamerV2>
        {
            new GamerV2 { Id = 1, FirstName = "NameFirstOne", LastName = "NameLastOne"},
            new GamerV2 { Id = 2, FirstName = "NameFirstTwo", LastName = "NameLastTwo"},
            new GamerV2 { Id = 3, FirstName = "NameFirstThree", LastName = "NameLastThree"}
        };

        public IEnumerable<GamerV2> Get()
        {
```

```
            return _gamers;
        }

        public GamerV2 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

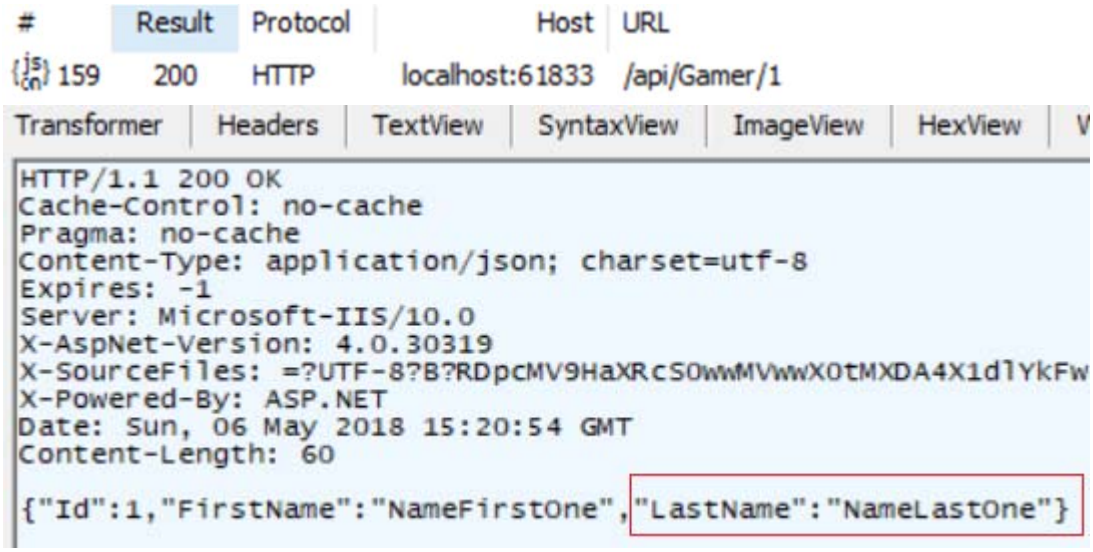
7. OnlineGame.WebApiD - Version by the request custom header property

7.1. What to do - Version by the request custom header property

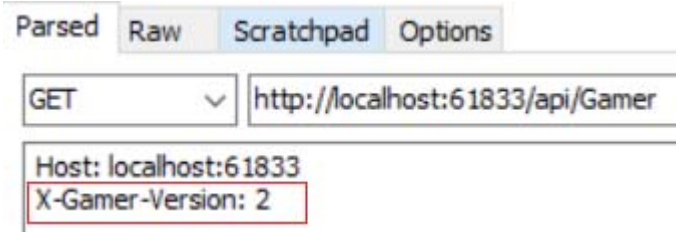
Reference:
http://csharp-video-tutorials.blogspot.com.au/2017/02/web-api-versioning-using-querystring.html
1.
api/Gamer
Call GamerV1Controller Get() action. List all gamers (version 1)
api/Gamer/1
Call GamerV1Controller Get(int id) action. Return id 1 gamer(version 1)
2.
Test in Fiddler
GET
http://localhost:61833/api/Gamer/1
api/Gamer/1
Request Header:
Host: localhost:61833
X-Gamer-Version: 2



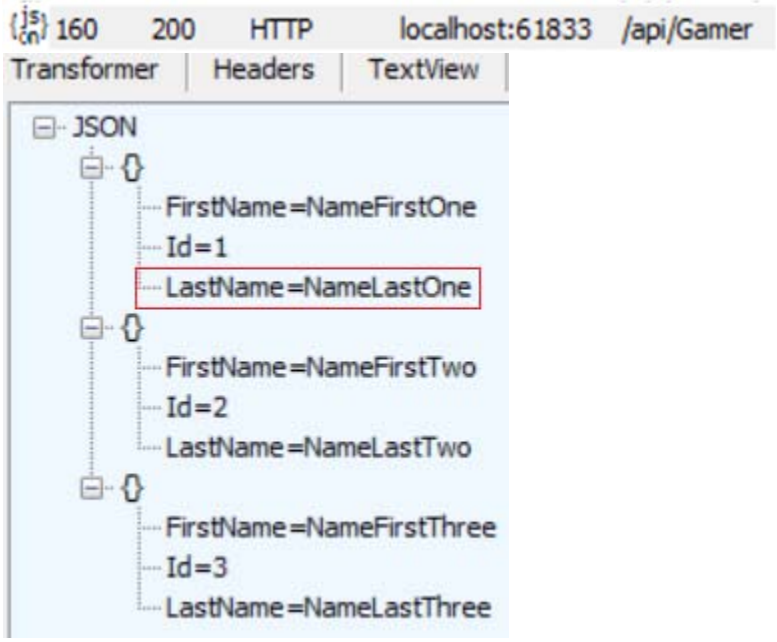
-->



3.
Test in Fiddler
GET
http://localhost:61833/api/Gamer
api/Gamer
Request Header:
Host: localhost:61833
X-Gamer-Version: 2



-->



-->

In Web API, **DefaultHttpControllerSelector** class **SelectController()** method selects the controller and action based on URI.
E.g.
/api/gamer/1
By default, **SelectController()** will takes "**gamer**" as the controller name and call "**GamerController**".
However, it does not fit our requirement.
We want when request header property "**X-Gamer-Version: 1**" call "**GamerV1Controller**".
We want when request header property "**X-Gamer-Version: 2**" call "**GamerV2Controller**".
Therefore, we need a "**CustomControllerSelector**"

7.2. OnlineGame.WebApiD/WebApiShare/CustomControllerSelector.cs - Version by the request custom header property

```
using System.Collections.Generic;
using System.Linq;
using  System.Net.Http;
using System.Web.Http;
using System.Web.Http.Controllers;
using System.Web.Http.Dispatcher;
```

```
using System.Web.Http.Routing;

namespace OnlineGame.WebApiD.WebApiShare
{
    public class CustomControllerSelector : DefaultHttpControllerSelector
    {
        private HttpConfiguration _configuration;

        public CustomControllerSelector(HttpConfiguration configuration) : base(configuration)
        {
            _configuration = configuration;
        }

        public override HttpControllerDescriptor SelectController(HttpRequestMessage request)
        {
            //1.
            //Get all API controllers
            // GetControllerMapping returns all controllers which extend ApiController
            IDictionary<string, HttpControllerDescriptor> controllers =
                GetControllerMapping();
            //request.GetRouteData() returns controller name and parameter values from the request URI
            IHttpRouteData routeData = request.GetRouteData();

            //2.
            //Get Controller Name
            // routeData.Values["controller"].ToString() returns
            // the controller name from route data.
            // In this case, the controller name is "Gamers".
            string controllerName =
                routeData.Values["controller"].ToString();

            //3.
            //Set default versionNumber
            // Default version number to 1
            string versionNumber = "1";


            //4.
            //Get the version number

            ////4.1.
            ////Get version value from QueryString value
            //NameValueCollection queryString =
            //    HttpUtility.ParseQueryString(request.RequestUri.Query);
            //if (queryString["v"] != null) versionNumber = queryString["v"];

            //4.2.
            //Get the version number from Custom version header
            //customHeader can be any string which we will use it when issuing a request.
            string customHeader = "X-Gamer-Version";
            if (request.Headers.Contains(customHeader))
                versionNumber = request.Headers.GetValues(customHeader).FirstOrDefault();


            //5.
```

```
            //Get the versionNumber from query string.
            // if versionNumber==1, then controllerName=controllerName+"V1"
            // if versionNumber==2, then controllerName=controllerName+"V2"
            controllerName =
                controllerName +
                (versionNumber == "1" ? "V1" : "V2");

            //6.
            //Find the Controller by the name
            HttpControllerDescriptor controllerDescriptor;
            if (controllers.TryGetValue(controllerName, out controllerDescriptor))
                return controllerDescriptor;
            return null;
        }
    }
}
```

7.3. OnlineGame.WebApiD/App_Start/WebApiConfig.cs - Version by the request custom header property

```
using System.Web.Http;
using System.Web.Http.Dispatcher;
using OnlineGame.WebApiC.WebApiShare;

namespace OnlineGame.WebApiD
{
    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 }
            );

            //Replace the default controller selector,IHttpControllerSelector,
            //with our custom controller selector,CustomControllerSelector.
            config.Services.Replace(typeof(IHttpControllerSelector),
                new CustomControllerSelector(config));
            }
    }
}
```

7.4. OnlineGame.WebApiD/Controllers/GamerV1Controller.cs - Version by the request custom header property

```
using System.Collections.Generic;
using System.Linq;
```

```
using System.Web.Http;
using OnlineGame.Data;

namespace OnlineGame.WebApiD.Controllers
{
    public class GamerV1Controller : ApiController
    {
        List<GamerV1> _gamers = new List<GamerV1>
        {
            new GamerV1 { Id = 1, Name = "NameOne"},
            new GamerV1 { Id = 2, Name = "NameTwo"},
            new GamerV1{ Id = 3, Name = "NameThree"},
        };

        public IEnumerable<GamerV1> Get()
        {
            return _gamers;
        }

        public GamerV1 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

7.5. OnlineGame.WebApiD/Controllers/GamerV2Controller.cs - Version by the request custom header property

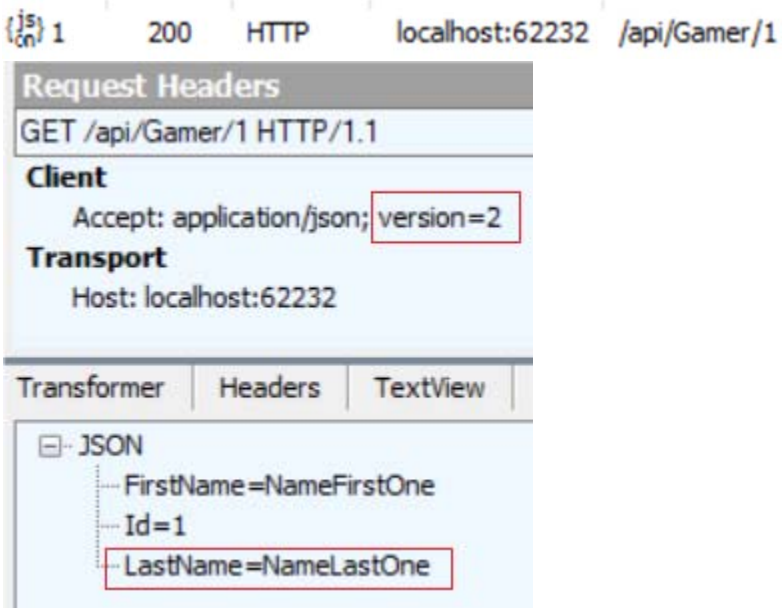
```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;
namespace OnlineGame.WebApiD.Controllers
{
    public class GamerV2Controller : ApiController
    {
        List<GamerV2> _gamers = new List<GamerV2>
        {
            new GamerV2 { Id = 1, FirstName = "NameFirstOne", LastName = "NameLastOne"},
            new GamerV2 { Id = 2, FirstName = "NameFirstTwo", LastName = "NameLastTwo"},
            new GamerV2 { Id = 3, FirstName = "NameFirstThree", LastName = "NameLastThree"}
        };

        public IEnumerable<GamerV2> Get()
        {
            return _gamers;
        }

        public GamerV2 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

8. OnlineGame.WebApiE - Version by the request header Accept property

8.1. What to do

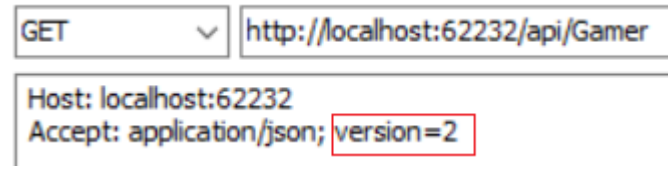
Reference:
http://csharp-video-tutorials.blogspot.com.au/2017/02/web-api-versioning-using-querystring.html
1.
api/Gamer
Call GamerV1Controller Get() action. List all gamers (version 1)
api/Gamer/1
Call GamerV1Controller Get(int id) action. Return id 1 gamer(version 1)
2.
Test in Fiddler
GET
http://localhost:62232/api/Gamer/1
api/Gamer/1
Request Header:
Host: localhost:62232
Accept: application/json; version=2

-->

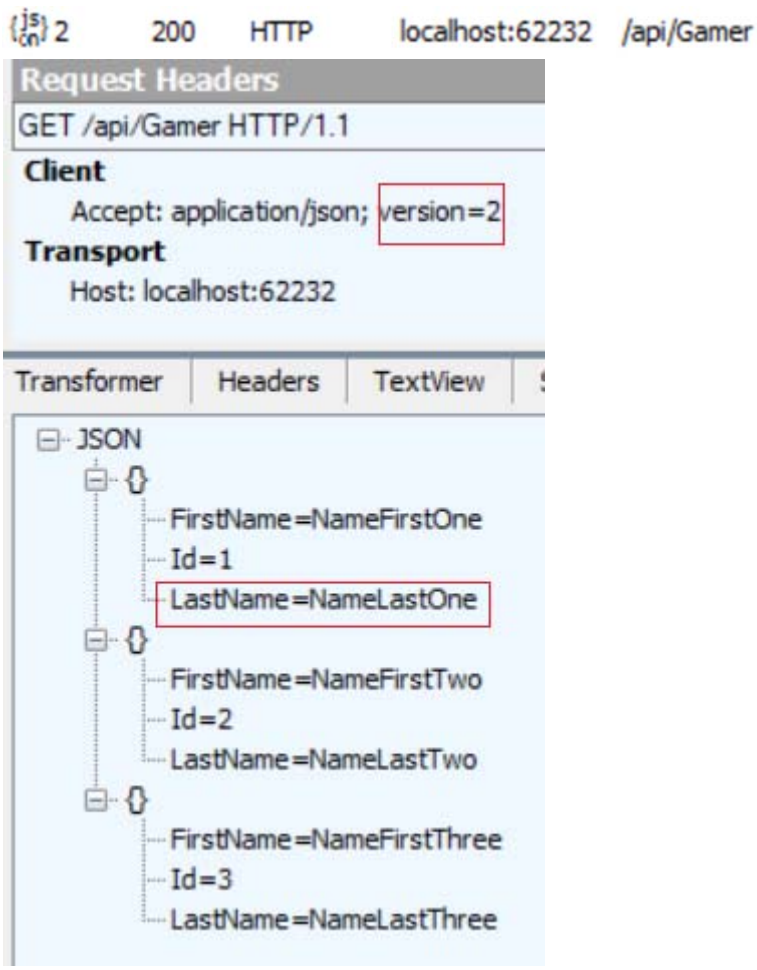


3.
Test in Fiddler
GET
http://localhost:62232/api/Gamer
api/Gamer
Request Header:
Host: localhost:62232
Accept: application/json; version=2

-->



-->



-->

In Web API, **DefaultHttpControllerSelector** class **SelectController()** method selects the controller and action based on URI.

E.g.

/api/gamer/1

By default, **SelectController()** will takes "**gamer**" as the controller name and call "**GamerController**".

However, it does not fit our requirement.

We want when request header accept property "**version=1**" call "**GamerV1Controller**".

We want when request header accept property "**version=2**" call "**GamerV2Controller**".

Therefore, we need a "**CustomControllerSelector**"

8.2. OnlineGame.WebApiE/WebApiShare/CustomControllerSelector.cs - Version by the request header Accept property

```
using System.Collections.Generic;
using System.Linq;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Web.Http;
using System.Web.Http.Controllers;
```

```
using System.Web.Http.Dispatcher;
using System.Web.Http.Routing;

namespace OnlineGame.WebApiE.WebApiShare
{
    public class CustomControllerSelector : DefaultHttpControllerSelector
    {
        private HttpConfiguration _configuration;

        public CustomControllerSelector(HttpConfiguration configuration) : base(configuration)
        {
            _configuration = configuration;
        }

        public override HttpControllerDescriptor SelectController(HttpRequestMessage request)
        {
            //1.
            //Get all API controllers
            // GetControllerMapping returns all controllers which extend ApiController
            IDictionary<string, HttpControllerDescriptor> controllers =
                GetControllerMapping();
            //request.GetRouteData() returns controller name and parameter values from the request URI
            IHttpRouteData routeData = request.GetRouteData();

            //2.
            //Get Controller Name
            // routeData.Values["controller"].ToString() returns
            // the controller name from route data.
            // In this case, the controller name is "Gamers".
            string controllerName =
                routeData.Values["controller"].ToString();

            //3.
            //Set default versionNumber
            // Default version number to 1
            string versionNumber = "1";


            //4. ------------------------------------
            //Get the version number

            ////4.1. ----------------
            ////Get version value from QueryString value
            //NameValueCollection queryString =
            //    HttpUtility.ParseQueryString(request.RequestUri.Query);
            //if (queryString["v"] != null) versionNumber = queryString["v"];

            ////4.2. ----------------
            ////Get the version number from Custom version header
            ////customHeader can be any string which we will use it when issuing a request.
            //string customHeader = "X-Gamer-Version";
            //if (request.Headers.Contains(customHeader))
            //    versionNumber = request.Headers.GetValues(customHeader).FirstOrDefault();
```

```
            //4.3. ----------------
            //Get the version number by the request header Accept property.
             //E.g. Accept: application/json; version=2

            //4.3.1.
            //request.Headers.Accept returns the value of request header accept property.
            //Request header Accept property can contains many parameters which are seperated by ";".
            //One of parameter can be "version", and we can read its value.
            //4.3.2.
            ////request.Headers.Accept.Where(a => a.Parameters.Count(p => p.Name.ToLower() == "version") >
0)
            //it tells us whether the Request header Accept property has the parameter called "version".
             IEnumerable<MediaTypeWithQualityHeaderValue> acceptHeader =
                 request.Headers.Accept.Where(a => a.Parameters
                                 .Count(p => p.Name.ToLower() == "version") > 0);

            //acceptHeader is possible to be multiple enumeration of IEnumerable,
            //thus, we have to convert to MediaTypeWithQualityHeaderValue[] or empty array.
             MediaTypeWithQualityHeaderValue[] mediaTypeWithQualityHeaderValues =
                 acceptHeader as MediaTypeWithQualityHeaderValue[] ?? acceptHeader.ToArray();

            // If the Request header Accept property has the parameter called "version".
            if (mediaTypeWithQualityHeaderValues.Any())
             {
                // Get the version parameter value from the Accept header
                 versionNumber = mediaTypeWithQualityHeaderValues.First().Parameters
                                 .First(p => p.Name.ToLower() == "version").Value;
             }



            //5. ---------------------------------------
            //Get the versionNumber from query string.
            // if versionNumber==1, then controllerName=controllerName+"V1"
            // if versionNumber==2, then controllerName=controllerName+"V2"
             controllerName =
                 controllerName +
                 (versionNumber == "1" ? "V1" : "V2");

            //6.
            //Find the Controller by the name
             HttpControllerDescriptor controllerDescriptor;
            if (controllers.TryGetValue(controllerName, out controllerDescriptor))
                return controllerDescriptor;
            return null;
        }
    }
}
```

8.3. OnlineGame.WebApiE/App_Start/WebApiConfig.cs - Version by the request header Accept property

```
using System.Web.Http;
using System.Web.Http.Dispatcher;
```

```
using OnlineGame.WebApiC.WebApiShare;

namespace OnlineGame.WebApiE
{
    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 }
            );

            //Replace the default controller selector,IHttpControllerSelector,
            //with our custom controller selector,CustomControllerSelector.
            config.Services.Replace(typeof(IHttpControllerSelector),
                new CustomControllerSelector(config));
            }
    }
}
```

8.4. OnlineGame.WebApiE/Controllers/GamerV1Controller.cs - Version by the request header Accept property

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;

namespace OnlineGame.WebApiE.Controllers
{
    public class GamerV1Controller : ApiController
    {
        List<GamerV1> _gamers = new List<GamerV1>
        {
            new GamerV1 { Id = 1, Name = "NameOne"},
            new GamerV1 { Id = 2, Name = "NameTwo"},
            new GamerV1{ Id = 3, Name = "NameThree"},
        };

        public IEnumerable<GamerV1> Get()
        {
            return _gamers;
        }

        public GamerV1 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

8.5. OnlineGame.WebApiE/Controllers/GamerV2Controller.cs - Version by the request header Accept property

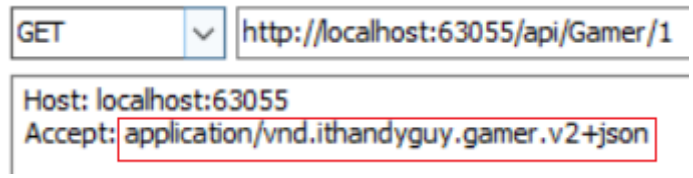
```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;
namespace OnlineGame.WebApiE.Controllers
{
    public class GamerV2Controller : ApiController
    {
        List<GamerV2> _gamers = new List<GamerV2>
        {
            new GamerV2 { Id = 1, FirstName = "NameFirstOne", LastName = "NameLastOne"},
            new GamerV2 { Id = 2, FirstName = "NameFirstTwo", LastName = "NameLastTwo"},
            new GamerV2 { Id = 3, FirstName = "NameFirstThree", LastName = "NameLastThree"}
        };

        public IEnumerable<GamerV2> Get()
        {
            return _gamers;
        }

        public GamerV2 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

9. OnlineGame.WebApiF - version by custom media types

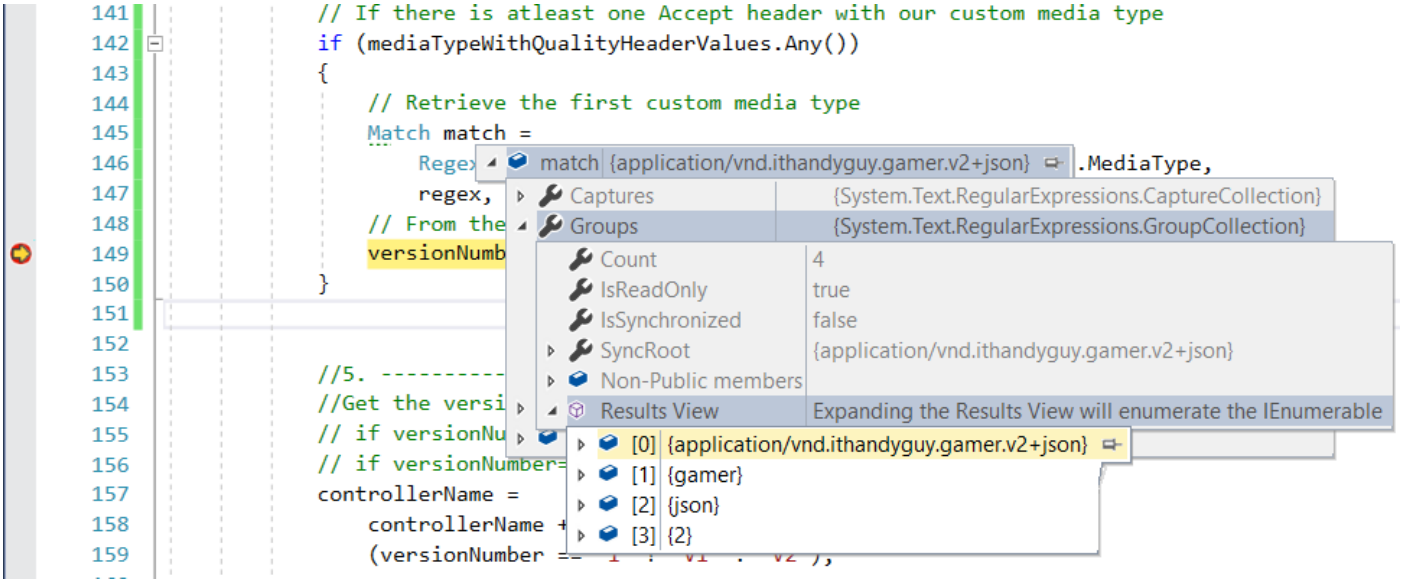
9.1. What to do - version by custom media types

Reference:
http://csharp-video-tutorials.blogspot.com.au/2017/02/web-api-versioning-using-querystring.html
1.
api/Gamer
Call GamerV1Controller Get() action. List all gamers (version 1)
api/Gamer/1
Call GamerV1Controller Get(int id) action. Return id 1 gamer(version 1)
2.
Test in Fiddler
GET
http://localhost:63055/api/Gamer/1
api/Gamer/1
Request Header:
Host: localhost:63055
Accept: **application/vnd.ithandyguy.gamer.v2+json**

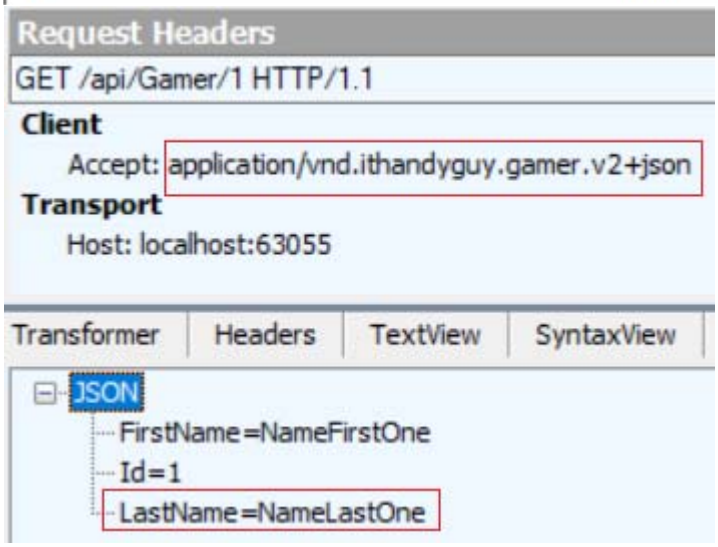
-->



-->



-->



3.

Test in Fiddler

GET

http://localhost:63055/api/Gamer/1

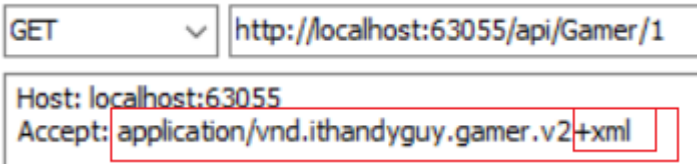
api/Gamer/1

Request Header:

Host: localhost:63055

Accept: **application/vnd.ithandyguy.gamer.v2+xml**

-->



-->

4.

Test in Fiddler

GET

http://localhost:63055/api/Gamer

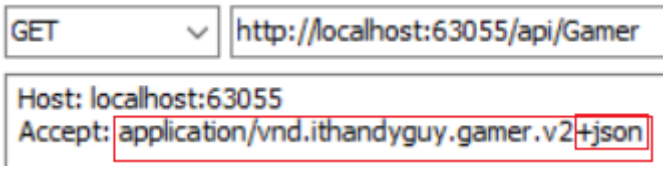
api/Gamer

Request Header:

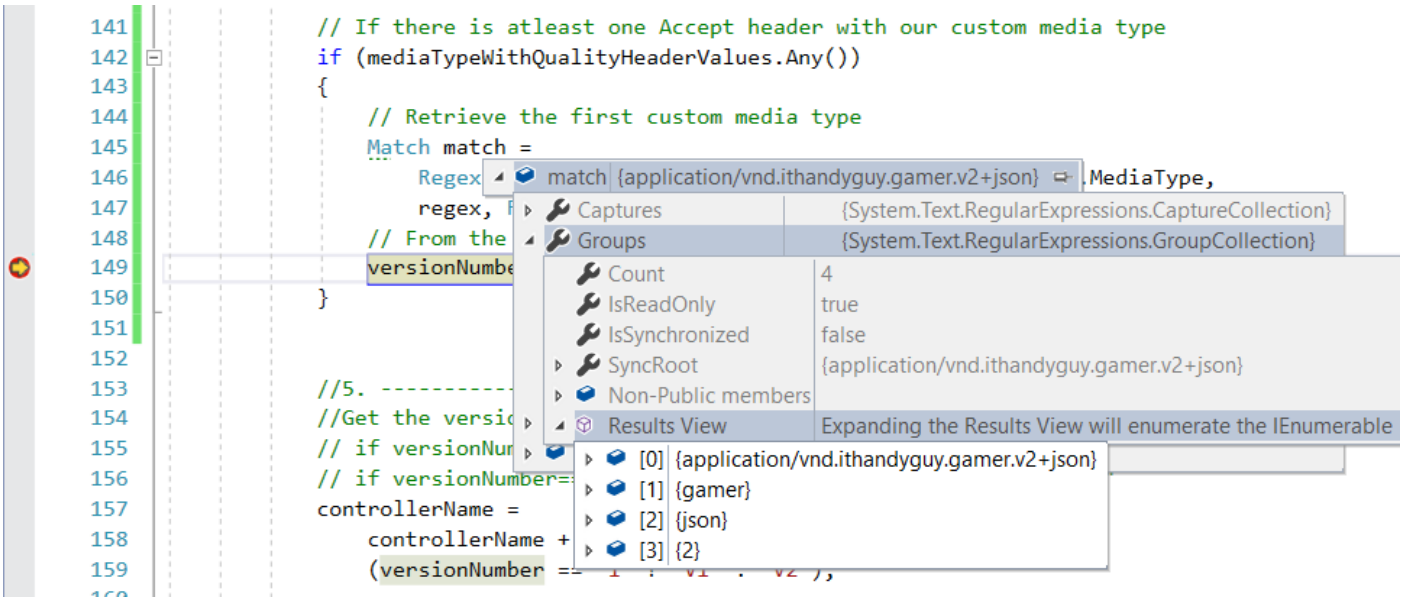
Host: localhost:63055

Accept: **application/vnd.ithandyguy.gamer.v2+json**

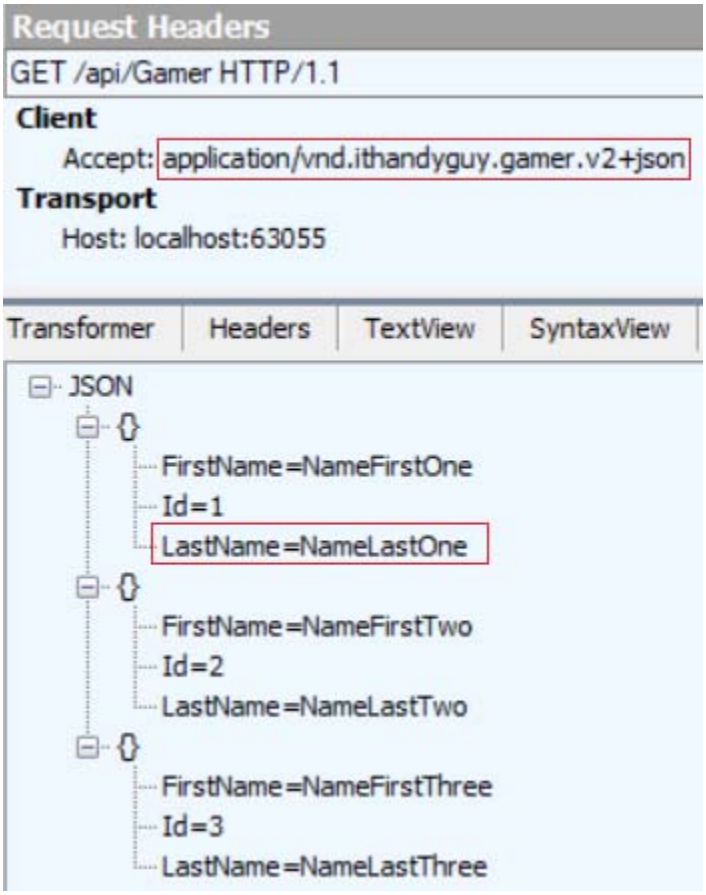
-->



-->



-->



5.

Test in Fiddler

GET

http://localhost:63055/api/Gamer

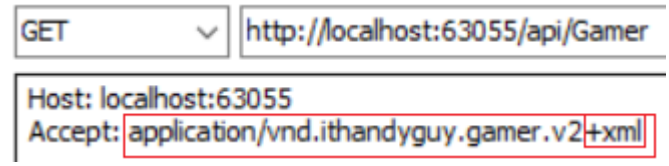
api/Gamer

Request Header:

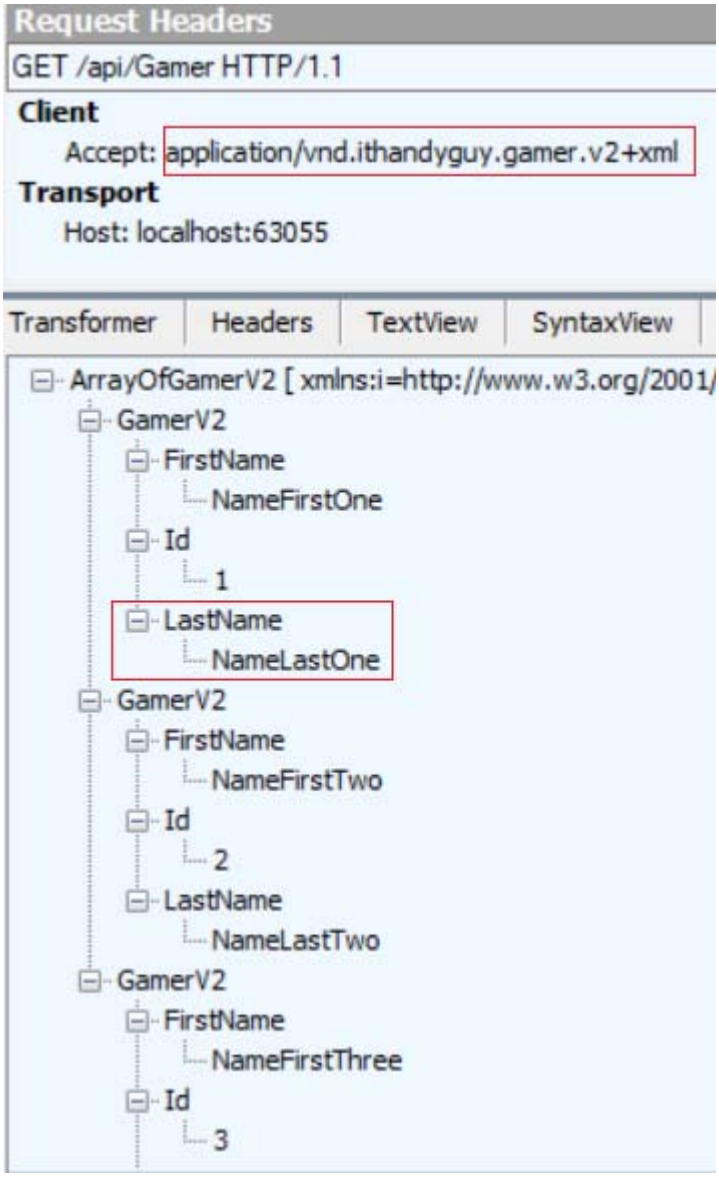
Host: localhost:63055

Accept: **application/vnd.ithandyguy.gamer.v2+xml**

-->



-->



9.2. OnlineGame.WebApiF/WebApiShare/CustomControllerSelector.cs - version by custom media types

```
using System.Collections.Generic;
using System.Linq;
using  System.Net.Http;
using  System.Net.Http.Headers;
using System.Text.RegularExpressions;
using System.Web.Http;
using System.Web.Http.Controllers;
using System.Web.Http.Dispatcher;
using System.Web.Http.Routing;

namespace OnlineGame.WebApiF.WebApiShare
{
    public class CustomControllerSelector : DefaultHttpControllerSelector
    {
        private HttpConfiguration _configuration;

        public CustomControllerSelector(HttpConfiguration configuration) : base(configuration)
        {
```

```
            _configuration = configuration;
        }

        public override HttpControllerDescriptor SelectController(HttpRequestMessage request)
        {
            //1.
            //Get all API controllers
            // GetControllerMapping returns all controllers which extend ApiController
            IDictionary<string, HttpControllerDescriptor> controllers =
                GetControllerMapping();
            //request.GetRouteData() returns controller name and parameter values from the request URI
            IHttpRouteData routeData = request.GetRouteData();

            //2.
            //Get Controller Name
            // routeData.Values["controller"].ToString() returns
            // the controller name from route data.
            // In this case, the controller name is "Gamers".
            string controllerName =
                routeData.Values["controller"].ToString();

            //3.
            //Set default versionNumber
            // Default version number to 1
            string versionNumber = "1";


            //4. ------------------------------------
            //Get the version number
            ////4.1. ----------------
            ////Get version value from QueryString value
            //NameValueCollection queryString =
            //    HttpUtility.ParseQueryString(request.RequestUri.Query);
            //if (queryString["v"] != null) versionNumber = queryString["v"];


            ////4.2. ----------------
            ////Get the version number from Custom version header
            ////customHeader can be any string which we will use it when issuing a request.
            //string customHeader = "X-Gamer-Version";
            //if (request.Headers.Contains(customHeader))
            //    versionNumber = request.Headers.GetValues(customHeader).FirstOrDefault();


            ////4.3. ----------------
            ////Get the version number by the request header Accept property.
            ////E.g. Accept: application/json; version=2
            ////4.3.1.
            ////request.Headers.Accept returns the value of request header accept property.
            ////Request header Accept property can contains many parameters which are seperated by ";".
            ////One of parameter can be "version", and we can read its value.
            ////4.3.2.
            //////request.Headers.Accept.Where(a => a.Parameters.Count(p => p.Name.ToLower() == "version")
> 0)
```

```
            ////it tells us whether the Request header Accept property has the parameter called "version".
            //IEnumerable<MediaTypeWithQualityHeaderValue> acceptHeader =
            //    request.Headers.Accept.Where(a => a.Parameters
            //                    .Count(p => p.Name.ToLower() == "version") > 0);

            ////acceptHeader is possible to be multiple enumeration of IEnumerable,
            ////thus, we have to convert to MediaTypeWithQualityHeaderValue[] or empty array.
            //MediaTypeWithQualityHeaderValue[] mediaTypeWithQualityHeaderValues =
            //    acceptHeader as MediaTypeWithQualityHeaderValue[] ?? acceptHeader.ToArray();

            //// If the Request header Accept property has the parameter called "version".
            //if (mediaTypeWithQualityHeaderValues.Any())
            //{
            //    // Get the version parameter value from the Accept header
            //    versionNumber = mediaTypeWithQualityHeaderValues.First().Parameters
            //                    .First(p => p.Name.ToLower() == "version").Value;
            //}


            ////4.4. ----------------
            //Get the version number from the Custom media type
            //4.4.1.
            //In request header "Accept" property.
            //E.g. Accept: applicaiton/xml   or  Accept: applicaiton/json
            //xml and json are media type.
            //We want to use Custom media type
            //E.g. Accept: applicaiton/vnd.ithandyguy.gamer.v1+json
            //In our case, it will call GamerV1Controller and return json format.
            //E.g. Accept: applicaiton/vnd.ithandyguy.gamer.v2+json
            //In our case, it will call GamerV2Controller and return json format.
            //4.4.2.
            //"vnd" means vendor specific media type
            //"vnd.ithandyguy" means vender ithandyguy
            //4.4.3.
            ////application\/vnd\.ithandyguy\.([a-z]+)\.v(?<version>[0-9]+)\+([a-z]+)
            //It is a regular expression.
            //E.g. "applicaiton/vnd.ithandyguy.gamer.v1+json"
            //4.4.3.1.
            //"application\/vnd\.ithandyguy\." means "application.vnd.ithandyguy"
            //4.4.3.2.
            //"([a-z]+)" means from a to z. "+" means any number of characters
            //4.4.3.3.
            //"[0-9]+" means from 0 to 9. "+" means any number of characters.
            //4.4.3.4.
            //"(?<version>[0-9]+)"
            //<version> is the group name.
            //using the group name("version") instead of ZERO based index
            //4.4.3.5.
            //versionNumber = match.Groups["version"].Value;
            //it retrieves the version number
            string regex =
                @"application\/vnd\.ithandyguy\.([a-z]+)\.v(?<version>[0-9]+)\+([a-z]+)";
```

```
            //Request Header Accept property contains many parameters.
            //It will check if any of parameter has our custom media type
            //by checking if there is a match with regular expression specified
            IEnumerable<MediaTypeWithQualityHeaderValue> acceptHeader =
                request.Headers.Accept
                .Where(a => Regex.IsMatch(a.MediaType, regex, RegexOptions.IgnoreCase));

            //acceptHeader is possible to be multiple enumeration of IEnumerable,
            //thus, we have to convert to MediaTypeWithQualityHeaderValue[] or empty array.
            MediaTypeWithQualityHeaderValue[] mediaTypeWithQualityHeaderValues =
                   acceptHeader as MediaTypeWithQualityHeaderValue[] ?? acceptHeader.ToArray();

            // If there is atleast one Accept header with our custom media type
            if (mediaTypeWithQualityHeaderValues.Any())
            {
                // Retrieve the first custom media type
                Match match =
                    Regex.Match(mediaTypeWithQualityHeaderValues.First().MediaType,
                    regex, RegexOptions.IgnoreCase);
                // From the version group, get the version number
                versionNumber = match.Groups["version"].Value;
            }



            //5. ----------------------------------------
            //Get the versionNumber from query string.
            // if versionNumber==1, then controllerName=controllerName+"V1"
            // if versionNumber==2, then controllerName=controllerName+"V2"
            controllerName =
                controllerName +
                (versionNumber == "1" ? "V1" : "V2");

            //6.
            //Find the Controller by the name
            HttpControllerDescriptor controllerDescriptor;
            if (controllers.TryGetValue(controllerName, out controllerDescriptor))
                return controllerDescriptor;
            return null;
        }
    }
}
```

9.3. OnlineGame.WebApiF/App_Start/WebApiConfig.cs - version by custom media types

```
using System.Web.Http;
using System.Web.Http.Dispatcher;
using OnlineGame.WebApiC.WebApiShare;

namespace OnlineGame.WebApiF
{
    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 }
             );

            //Replace the default controller selector,IHttpControllerSelector,
            //with our custom controller selector,CustomControllerSelector.
             config.Services.Replace(typeof(IHttpControllerSelector),
                 new CustomControllerSelector(config));
             }

             //Add custom media type for JSON formatter
             config.Formatters.JsonFormatter.SupportedMediaTypes
                 .Add(new MediaTypeHeaderValue("application/vnd.ithandyguy.gamer.v1+json"));
             config.Formatters.JsonFormatter.SupportedMediaTypes
                 .Add(new MediaTypeHeaderValue("applicaiton/vnd.ithandyguy.gamer.v2+json"));

            //Add custom media type for XML formatter
             config.Formatters.XmlFormatter.SupportedMediaTypes
                 .Add(new MediaTypeHeaderValue("application/vnd.ithandyguy.gamer.v1+xml"));
             config.Formatters.XmlFormatter.SupportedMediaTypes
                 .Add(new MediaTypeHeaderValue("application/vnd.ithandyguy.gamer.v2+xml"));
    }
}
```

9.4. OnlineGame.WebApiF/Controllers/GamerV1Controller.cs - version by custom media types

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;

namespace OnlineGame.WebApiF.Controllers
{
    public class GamerV1Controller : ApiController
    {
        List<GamerV1> _gamers = new List<GamerV1>
        {
            new GamerV1 { Id = 1, Name = "NameOne"},
            new GamerV1 { Id = 2, Name = "NameTwo"},
            new GamerV1{ Id = 3, Name = "NameThree"},
        };

        public IEnumerable<GamerV1> Get()
        {
            return _gamers;
        }

        public GamerV1 Get(int id)
```

```
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```

9.5. OnlineGame.WebApiF/Controllers/GamerV2Controller.cs - version by custom media types

```
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using OnlineGame.Data;
namespace OnlineGame.WebApiF.Controllers
{
    public class GamerV2Controller : ApiController
    {
        List<GamerV2> _gamers = new List<GamerV2>
        {
            new GamerV2 { Id = 1, FirstName = "NameFirstOne", LastName = "NameLastOne"},
            new GamerV2 { Id = 2, FirstName = "NameFirstTwo", LastName = "NameLastTwo"},
            new GamerV2 { Id = 3, FirstName = "NameFirstThree", LastName = "NameLastThree"}
        };

        public IEnumerable<GamerV2> Get()
        {
            return _gamers;
        }

        public GamerV2 Get(int id)
        {
            return _gamers.FirstOrDefault(s => s.Id == id);
        }
    }
}
```