In this video we will discuss the use of **RoutePrefix attribute** with an example. This is continuation to [Part 31](http://csharp-video-tutorials.blogspot.com/2017/02/attribute-routing-in-aspnet-web-api-2.html). Please watch [Part 31](http://csharp-video-tutorials.blogspot.com/2017/02/attribute-routing-in-aspnet-web-api-2.html) from [ASP.NET Web API tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b) before proceeding.   
  
  
  
**RoutePrefix attribute :** As you can see from the example below, all the routes in the **StudentsController** start with the same prefix - api/students

public class StudentsController : ApiController

{

    [Route("api/students")]

    public IEnumerable<Student> Get()

    [Route("api/students/{id}")]

    public Student Get(int id)

    [Route("api/students/{id}/courses")]

    public IEnumerable<string> GetStudentCourses(int id)

}

The common prefix "api/students" can be specified for the entire controller using the [RoutePrefix] attribute as shown below. This eliminates the need to repeat the common prefix "api/students" on every controller action method.

[RoutePrefix("api/students")]

public class StudentsController : ApiController

{

    [Route("api/students")]

    public IEnumerable<Student> Get()

    [Route("api/students/{id}")]

    public Student Get(int id)

    [Route("api/students/{id}/courses")]

    public IEnumerable<string> GetStudentCourses(int id)

}

However, sometimes you may want to override the route prefix. Let us understand this with an example.  
  
Right click on the models folder, and add a new class file. Name it **"Teacher.cs"**. Copy and paste the following code.

namespace WebAPI.Models

{

    public class Teacher

    {

        public int Id { get; set; }

        public string Name { get; set; }

    }

}

Add the following **GetTeachers()** method to the **"StudentsController"**.

public IEnumerable<Teacher> GetTeachers()

{

    List<Teacher> teachers = new List<Teacher>()

    {

        new Teacher() { Id = 1, Name = "Rob" },

                new Teacher() { Id = 2, Name = "Mike" },

        new Teacher() { Id = 3, Name = "Mary" }

    };

    return teachers;

}

So at this point, **"StudentsController"** class is as shown below

[RoutePrefix("api/students")]

public class StudentsController : ApiController

{

    static List<Student> students = new List<Student>()

    {

        new Student() { Id = 1, Name = "Tom" },

        new Student() { Id = 2, Name = "Sam" },

        new Student() { Id = 3, Name = "John" }

    };

    [Route("")]

    public IEnumerable<Student> Get()

    {

        return students;

    }

    [Route("{id}")]

    public Student Get(int id)

    {

        return students.FirstOrDefault(s => s.Id == id);

    }

    [Route("{id}/courses")]

    public IEnumerable<string> GetStudentCourses(int id)

    {

        if (id == 1)

            return new List<string>() { "C#", "ASP.NET", "SQL Server" };

        else if (id == 2)

            return new List<string>() { "ASP.NET Web API", "C#", "SQL Server" };

        else

            return new List<string>() { "Bootstrap", "jQuery", "AngularJs" };

    }

    public IEnumerable<Teacher> GetTeachers()

    {

        List<Teacher> teachers = new List<Teacher>()

    {

        new Teacher() { Id = 1, Name = "Rob" },

        new Teacher() { Id = 2, Name = "Mike" },

        new Teacher() { Id = 3, Name = "Mary" }

    };

        return teachers;

    }

}

We want **GetTeachers()** method to be mapped to URI "/api/teachers".  

[Route("api/teachers")]

public IEnumerable<Teacher> GetTeachers()

{

    List<Teacher> teachers = new List<Teacher>()

    {

        new Teacher() { Id = 1, Name = "Rob" },

        new Teacher() { Id = 2, Name = "Mike" },

        new Teacher() { Id = 3, Name = "Mary" }

    };

    return teachers;

}

If we use the [Route] attribute on **GetTeachers()** method as shown above and when we navigate to /api/teachers, we get the following error.  
No HTTP resource was found that matches the request URI 'http://localhost:65116/api/teachers'.  
  
But if we navigate to /api/students/api/teachers then we get the list of teachers. This is because of the [RoutePrefix("api/students")] attribute on StudentsController. So there is definitely a need to override the RoutePrefix used on the StudentsController. To override the RoutePrefix use **~** as shown below

[Route("~/api/teachers")]

public IEnumerable<Teacher> GetTeachers()

{

    List<Teacher> teachers = new List<Teacher>()

    {

        new Teacher() { Id = 1, Name = "Rob" },

        new Teacher() { Id = 2, Name = "Mike" },

        new Teacher() { Id = 3, Name = "Mary" }

    };

    return teachers;

}

With this change GetTeachers() action method is mapped to URI "/api/teachers" as expected.  
  
**What is the use of RoutePrefix attribute**  
RoutePrefix attribute is used to specify the common route prefix at the controller level to eliminate the need to repeat that common route prefix on every controller action method  
  
**How to override the route prefix**  
Use ~ character to override the route prefix