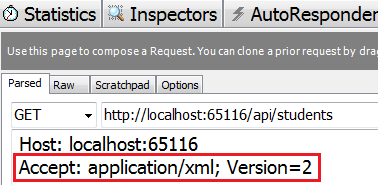
**In this video we will discuss**  
1. What is Accept header  
2. Versioning a Web API Service using the standard accept header    
  
This is continuation to [Part 37](http://csharp-video-tutorials.blogspot.com/2017/03/web-api-versioning-using-custom-header.html). Please watch [Part 37](http://csharp-video-tutorials.blogspot.com/2017/03/web-api-versioning-using-custom-header.html) from [Web API tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b) before proceeding.  
  
  
  
**What is Accept header**  
The Accept header tells the server in what file format the browser wants the data. These file formats are more commonly called as MIME-types. MIME stands for Multipurpose Internet Mail Extensions.  
  
  
  
In [Part 5](http://csharp-video-tutorials.blogspot.com/2016/09/aspnet-web-api-content-negotiation.html) of this [Web API tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b) we discussed that, if you want the 

* Web API service to return data in XML format you set the Accept header to application/xml
* Web API service to return data in JSON format you set the Accept header to application/json

**Versioning a Web API Service using the accept header**  
Instead of creating a custom header, just for versioning purpose, we can use the standard Accept header. We can add parameters to the Accept header to send any additional data along with the request to the server. For example, we can specify the version of the service we want using the version parameter as shown below.   
  
   
  
All we have to do now is, read the version parameter value from the Accept header in our CustomControllerSelector class. The changes that are required are commented, so it is self-explanatory.

using System.Linq;

using System.Net.Http;

using System.Web.Http;

using System.Web.Http.Controllers;

using System.Web.Http.Dispatcher;

namespace WebAPI.Custom

{

    public class CustomControllerSelector : DefaultHttpControllerSelector

    {

        private HttpConfiguration \_config;

        public CustomControllerSelector(HttpConfiguration config) : base(config)

        {

            \_config = config;

        }

        public override HttpControllerDescriptor

            SelectController(HttpRequestMessage request)

        {

            var controllers = GetControllerMapping();

            var routeData = request.GetRouteData();

            var controllerName = routeData.Values["controller"].ToString();

            string versionNumber = "1";

            // Comment the code that gets the version number from Query String

            //var versionQueryString = HttpUtility.ParseQueryString(request.RequestUri.Query);

            //if (versionQueryString["v"] != null)

            //{

            //    versionNumber = versionQueryString["v"];

            //}

            // Get the version number from Custom version header

            //string customHeader = "X-StudentService-Version";

            //if (request.Headers.Contains(customHeader))

            //{

            //    // If X-StudentService-Version:1 is specified twice in the request

            //    // then in versionNumber variable will get a value of "1,1"

            //    versionNumber = request.Headers.GetValues(customHeader).FirstOrDefault();

            //    // Check if versionNumber string contains a comma, and take only

            //    // the first number from the comma separated list of version numbers

            //    if (versionNumber.Contains(","))

            //    {

            //        versionNumber = versionNumber.Substring(0, versionNumber.IndexOf(","));

            //    }

            //}

            // Get the version number from the Accept header

            // Users can include multiple Accept headers in the request

            // Check if any of the Accept headers has a parameter with name version

            var acceptHeader = request.Headers.Accept.Where(a => a.Parameters

                                .Count(p => p.Name.ToLower() == "version") > 0);

            // If there is atleast one header with a "version" parameter

            if (acceptHeader.Any())

            {

                // Get the version parameter value from the Accept header

                versionNumber = acceptHeader.First().Parameters

                                .First(p => p.Name.ToLower() == "version").Value;

            }

            HttpControllerDescriptor controllerDescriptor;

            if (versionNumber == "1")

            {

                controllerName = string.Concat(controllerName, "V1");

            }

            else

            {

                controllerName = string.Concat(controllerName, "V2");

            }

            if (controllers.TryGetValue(controllerName, out controllerDescriptor))

            {

                return controllerDescriptor;

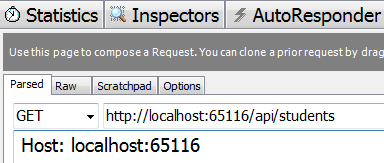
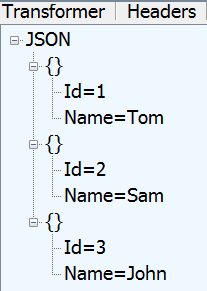
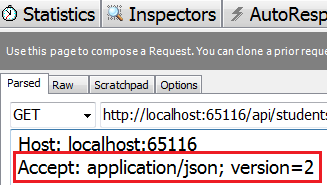
            }

            return null;

        }

    }

}

With the above changes in place, if we issue a request without the Accept header from fiddler, the service falls back to version 1 and returns version 1 student objects in JSON format as expected.   
  
   
  
Response from the Web API service  
   
  
On the other hand if we specify the version parameter as part of the Accept header, we get the specified version as expected.   
  
   
  
Response from the Web API service   
  
