**In this video we will discuss**

* What is MediaTypeFormatter
* How to return only JSON from ASP.NET Web API Service irrespective of the Accept header value
* How to return only XML from ASP.NET Web API Service irrespective of the Accept header value
* How to return JSON instead of XML from ASP.NET Web API Service when a request is made from the browser
* Point you to an article that describes how to return CSV formatted data from ASP.NET Web API

**What is MediaTypeFormatter**  
MediaTypeFormatter is an abstract class from which JsonMediaTypeFormatter and XmlMediaTypeFormatter classes inherit from. JsonMediaTypeFormatter handles JSON and XmlMediaTypeFormatter handles XML.  
  
**How to return only JSON from ASP.NET Web API Service irrespective of the Accept header value**  
Include the following line in **Register**() method of WebApiConfig.cs file in **App\_Start**folder. This line of code completely removes XmlFormatter which forces ASP.NET Web API to always return JSON irrespective of the Accept header value in the client request. Use this technique when you want your service to support only JSON and not XML.   
  
With this change, irrespective of the Accept header value (application/xml or application/json), the Web API service is always going to return JSON.

config.Formatters.Remove(config.Formatters.XmlFormatter);

**How to return only XML from ASP.NET Web API Service irrespective of the Accept header value**  
Include the following line in Register() method of WebApiConfig.cs file in App\_Start folder. This line of code completely removes JsonFormatter which forces ASP.NET Web API to always return XML irrespective of the Accept header value in the client request. Use this technique when you want your service to support only XML and not JSON.  
  
config.Formatters.Remove(config.Formatters.JsonFormatter);  
  
With this change, irrespective of the Accept header value (application/xml or application/json), the Web API service is always going to return XML.  
  
**How to return JSON instead of XML from ASP.NET Web API Service when a request is made from the browser.**  
So here is what we want the service to do  
1. When a request is issued from the browser, the web API service should return JSON instead of XML.   
2. When a request is issued from a tool like fiddler the Accept header value should be respected. This means if the Accept header is set to application/xml the service should return XML and if it is set to application/json the service should return JSON.  
  
There are 2 ways to achieve this  
  
**Approach 1 :**Include the following line in Register() method of WebApiConfig.cs file in App\_Start folder. This tells ASP.NET Web API to use JsonFormatter when a request is made for text/html which is the default for most browsers. The problem with this approach is that Content-Type header of the response is set to text/html which is misleading.

config.Formatters.JsonFormatter.SupportedMediaTypes

    .Add(new MediaTypeHeaderValue("text/html"));

**Approach 2 :** Include the following class in WebApiConfig.cs file in App\_Start folder. 

public class CustomJsonFormatter : JsonMediaTypeFormatter

{

    public CustomJsonFormatter()

    {

        this.SupportedMediaTypes.Add(new MediaTypeHeaderValue("text/html"));

    }

    public override void SetDefaultContentHeaders(Type type, HttpContentHeadersheaders, MediaTypeHeaderValue mediaType)

    {

        base.SetDefaultContentHeaders(type, headers, mediaType);

        headers.ContentType = new MediaTypeHeaderValue("application/json");

    }

}

**Register the formatter:** Place the following line in Register() method of WebApiConfig.cs file in App\_Start folder  
config.Formatters.Add(new CustomJsonFormatter());  
  
With these 2 changes, when a request is issued from the browser you will get JSON formatted data and the Content-Type header of the response is also set to application/json. If you are using tools like fiddler and if you set Accept header to application/xml you will still get XML formatted data.  
  
ASP.NET Web API is an extinsible framework. This means you can also plugin your own custom formatter. For example, if you want the response to be in CSV format, you can create custom CSVMediaTypeFormatter that inherits from the base abstract class MediaTypeFormatter . The following article describes how to do this.  
<http://www.tugberkugurlu.com/archive/creating-custom-csvmediatypeformatter-in-asp-net-web-api-for-comma-separated-values-csv-format>