**1) What is Web API?**

It is a framework which helps us to build/develop HTTP services. So there will a client server communication using HTTP protocol.

**2) What is Representational state transfer or REST?**

REST is architectural style, which has defined guidelines for creating services which are scalable. REST used with HTTP protocol using its verbs GET, POST, PUT and DELETE.

**3) Explain Web API Routing?**

Routing is the mechanism of pattern matching as we have in MVC. These routes will get registered in Route Tables. Below is the sample route in Web API –

Routes.MapHttpRoute(  
 Name: "MyFirstWebAPIRoute",  
 routeTemplate: “api/{controller}/{id}  
 defaults: new { id = RouteParameter.Optional}  
};

**4) List out the differences between WCF and Web API?**

WCF

* It is framework build for building or developing service oriented applications.
* WCF can be consumed by clients which can understand XML.
* WCF supports protocols like – HTTP, TCP, Named Pipes etc.

Web API

* It is a framework which helps us to build/develop HTTP services
* Web API is an open source platform.
* It supports most of the MVC features which keep Web API over WCF.

**5) What are the advantages of using REST in Web API?**

REST always used to make less data transfers between client and server which makes REST an ideal for using it in mobile apps. Web API supports HTTP protocol thereby it reintroduces the old way of HTTP verbs for communication.

**6) Difference between WCF Rest and Web API?**

WCF Rest

* “WebHttpBinding” to be enabled for WCF Rest.
* For each method there has to be attributes like – “WebGet” and “WebInvoke”
* For GET and POST verbs respectively.

Web API

* Unlike WCF Rest we can use full features of HTTP in Web API.
* Web API can be hosted in IIS or in application.

**7) List out differences between MVC and Web API?**

Below are some of the differences between MVC and Web API

MVC

* MVC is used to create a web app, in which we can build web pages.
* For JSON it will return JSONResult from action method.
* All requests are mapped to the respective action methods.

Web API

* This is used to create a service using HTTP verbs.
* This returns XML or JSON to client.
* All requests are mapped to actions using HTTP verbs.

**8) What are the advantages of Web API?**

Below are the list of support given by Web API –

* OData
* Filters
* Content Negotiation
* Self Hosting
* Routing
* Model Bindings

**9) Can we unit test Web API?**

Yes we can unit test Web API.

**10) How to unit test Web API?**

We can unit test the Web API using Fiddler tool. Below are the settings to be done in Fiddler –

*Compose Tab -> Enter Request Headers -> Enter the Request Body and execute*

**11) Can we return view from Web API?**

No. We cannot return view from Web API.

**12) How we can restrict access to methods with specific HTTP verbs in Web API?**

Attribute programming is used for this functionality. Web API will support to restrict access of calling methods with specific HTTP verbs. We can define HTTP verbs as attribute over method as shown below

[HttpPost]  
 public void UpdateTestCustomer(Customer c)  
 {  
 TestCustomerRepository.AddCustomer(c);   
 }

**13) Can we use Web API with ASP.NET Web Forms?**

Yes. We can use Web API with ASP.NET Webforms.

**14) List out the steps to be made for Web API to work in Web Forms?**

Below are the steps to be followed –

* Creating new controller for Web API.
* Adding routing table to “Application\_Start” method in Global.asax
* Make a AJAX call to Web API actions.

**15) Explain how to give alias name for action methods in Web API?**

Using attribute “ActionName” we can give alias name for Web API actions. Eg:

[HttpPost]  
[ActionName("AliasTestAction")]  
 public void UpdateTestCustomer(Customer c)  
 {  
 TestCustomerRepository.AddCustomer(c);   
 }

**16) What is the difference between MVC Routing and Web API Routing?**

There should be atleast one route defined for MVC and Web API to run MVC and Web API application respectively. In Web API pattern we can find “api/” at the beginning which makes it distinct from MVC routing. In Web API routing “action” parameter is not mandatory but it can be a part of routing.

**17) Explain Exception Filters?**

Exception filters will be executed whenever controller methods (actions) throws an exception which is unhandled. Exception filters will implement “IExceptionFilter” interface.

**18) Explain about the new features added in Web API 2.0 version?**

Below are the list of features introduced in Web API 2.0 –

* OWIN
* Attribute Routing
* External Authentication
* Web API OData

**19) How can we pass multiple complex types in Web API?**

Below are the methods to pass the complex types in Web API –

* Using ArrayList
* Newtonsoft JArray

**20) Write a code snippet for passing arraylist in Web API?**

Below is the code snippet for passing arraylist –

ArrayList paramList = new ArrayList();

Category c = new Category { CategoryId = 1, CategoryName = "SmartPhones"};  
Product p = new Product { ProductId = 1, Name = "Iphone", Price = 500, CategoryID = 1 };  
   
paramList.Add(c);  
paramList.Add(p);

**21) Give an example of Web API Routing?**

Below is the sample code snippet to show Web API Routing –

config.Routes.MapHttpRoute(  
name: "MyRoute",//route name  
routeTemplate: "api/{controller}/{action}/{id}",//as you can see "api" is at the beginning.  
defaults: new { id = RouteParameter.Optional }  
);

**22) Give an example of MVC Routing?**

Below is the sample code snippet to show MVC Routing –

routes.MapRoute(  
 name: "MyRoute", //route name  
 url: "{controller}/{action}/{id}", //route pattern  
 defaults: new   
 {   
 controller = "a4academicsController",   
 action = "a4academicsAction",   
 id = UrlParameter.Optional  
 }   
 );

**23) How we can handle errors in Web API?**

Below are the list of classes which can be used for error handling -

* HttpResponseException
* Exception Filters
* Registering Exception Filters
* HttpError

**24) Explain how we can handle error from “HttpResponseException”?**

This returns the HTTP status code what you specify in the constructor. Eg :

public TestClass MyTestAction(int id)  
{  
 TestClass c = repository.Get(id);  
 if (c == null)  
 {  
 throw new HttpResponseException(HttpStatusCode.NotFound);  
 }  
 return c;  
}

**25) How to register Web API exception filters?**

Below are the options to register Web API exception filters –

* From Action
* From Controller
* Global registration

**26) Write a code snippet to register exception filters from action?**

Below is the code snippet for registering exception filters from action –

[NotImplExceptionFilter]  
public TestCustomer GetMyTestCustomer(int custid)  
{  
 //Your code goes here  
}

**27) Write a code snippet to register exception filters from controller?**

Below is the code snippet for registering exception filters from controller –

[NotImplExceptionFilter]  
public class TestCustomerController : Controller  
{  
 //Your code goes here  
}

**28) Write a code snippet to register exception filters globally?**

Below is the code snippet for registering exception filters globally –

GlobalConfiguration.Configuration.Filters.Add( new MyTestCustomerStore.NotImplExceptionFilterAttribute());

**29) How to handle error using HttpError?**

HttpError will be used to throw the error info in response body. “CreateErrorResponse” method is used along with this, which is an extension method defined in “HttpRequestMessageExtensions”.

**30) Write a code snippet to show how we can return 404 error from HttpError?**

Below is the code snippet for returning 404 error from HttpError –

string message = string.Format("TestCustomer id = {0} not found", customerid);  
return Request.CreateErrorResponse(HttpStatusCode.NotFound, message);

**31) How to enable tracing in Web API?**

To enable tracing place below code in –“Register” method of WebAPIConfig.cs file.

config.EnableSystemDiagnosticsTracing();

**32) Explain how Web API tracing works?**

Tracing in Web API done in façade pattern i.e, when tracing for Web API is enabled, Web API will wrap different parts of request pipeline with classes, which performs trace calls.

**33) Can we unit test Web API?**

Yes we can unit test Web API.

**34) Explain Authentication in Web API?**

Web API authentication will happen in host. In case of IIS it uses Http Modules for authentication or we can write custom Http Modules. When host is used for authentication it used to create principal, which represent security context of the application.

**35) Explain ASP.NET Identity?**

This is the new membership system for ASP.NET. This allows to add features of login in our application.

Below are the list of features supported by ASP.NET Identity in Web API –

* One ASP.NET Identity System
* Persistence Control

**36) What are Authentication Filters in Web API?**

Authentication Filter will let you set the authentication scheme for actions or controllers. So this way our application can support various authentication mechanisms.

**37) How to set the Authentication filters in Web API?**

Authentication filters can be applied at the controller or action level. Decorate attribute – "IdentityBasicAuthentication” over controller where we have to set the authentication filter.

**38) Explain method – “AuthenticateAsync” in Web API?**

“AuthenticateAsync” method will create “IPrincipal” and will set on request. Below is the sample code snippet for “AuthenticateAsync” –

Task AuthenticateAsync(  
 HttpAuthenticationContext mytestcontext,  
 CancellationToken mytestcancellationToken  
)

**39) How to set the Error Result in Web API?**

Below is the sample code to show how to set error result in Web API –

HttpResponseMessage myresponse = new HttpResponseMessage(HttpStatusCode.Unauthorized);  
myresponse.RequestMessage = Request;  
myresponse.ReasonPhrase = ReasonPhrase;

**40) Explain method – “ChallengeAsync” in Web API?**

“ChallengeAsync” method is used to add authentication challenges to response. Below is the method signature –

Task ChallengeAsync(  
 HttpAuthenticationChallengeContext mytestcontext,  
 CancellationToken mytestcancellationToken  
)

**41) What are media types?**

It is also called MIME, which is used to identify the data . In Html, media types is used to describe message format in the body.

**42) List out few media types of HTTP?**

Below are the list of media types –

* Image/Png
* Text/HTML
* Application/Json

**43) Explain Media Formatters in Web API?**

Media Formatters in Web API can be used to read the CLR object from our HTTP body and Media formatters are also used for writing CLR objects of message body of HTTP.

**44) How to serialize read-only properties?**

Read-Only properties can be serialized in Web API by setting the value “true” to the property –

“SerializeReadOnlyTypes” of class – “DataContractSerializerSettings”.

**45) How to get Microsoft JSON date format ?**

Use “DateFormatHandling” property in serializer settings as below –

var myjson = GlobalConfiguration.Configuration.Formatters.JsonFormatter;  
myjson.SerializerSettings.DateFormatHandling = Newtonsoft.Json.DateFormatHandling.MicrosoftDateFormat;

**46) How to indent the JSON in web API?**

Below is the code snippet to make JSON indenting –

var mytestjson = GlobalConfiguration.Configuration.Formatters.JsonFormatter;  
mytestjson.SerializerSettings.Formatting = Newtonsoft.Json.Formatting.Indented;

**47) How to JSON serialize anonymous and weakly types objects?**

Using “Newtonsoft.Json.Linq.JObject” we can serialize and deserialize weakly typed objects.

**48) What is the use of “IgnoreDataMember” in Web API?**

By default if the properties are public then those can be serialized and deserialized, if we does not want to serialize the property then decorate the property with this attribute.

**49) How to write indented XML in Web API?**

To write the indented xml set “Indent” property to true.

**50) How to set Per-Type xml serializer?**

We can use method – “SetSerializer”. Below is the sample code snippet for using it –

var mytestxml = GlobalConfiguration.Configuration.Formatters.XmlFormatter;  
// Use XmlSerializer for instances of type "Product":  
mytestxml.SetSerializer<Product>(new XmlSerializer(typeof(MyTestCustomer)));

**51) What is “Under-Posting” and “Over-Posting” in Web API?**

* “Under-Posting” - When client leaves out some of the properties while binding then it’s called under – posting.
* “Over-Posting” – If the client sends more data than expected in binding then it’s called over-posting.

**52) How to handle validation errors in Web API?**

Web API will not return error to client automatically on validation failure. So its controller’s duty to check the model state and response to that. We can create a custom action filter for handling the same.

**53) Give an example of creating custom action filter in Web API?**

Below is the sample code for creating custom action filter –

public class MyCustomModelAttribute : ActionFilterAttribute  
{  
 public override void OnActionExecuting(HttpActionContext actionContext)  
 {  
 if (actionContext.ModelState.IsValid == false)  
 {  
 //Code goes here  
 }  
 }  
}

In case validation fails here it returns HTTP response which contains validation errors.

**54) How to apply custom action filter in WebAPI.config?**

Add a new action filter in “Register” method as shown -

public static class WebApiConfig  
 {  
 public static void Register(HttpConfiguration config)  
 {  
 config.Filters.Add(new MyCustomModelAttribute());  
// ...  
 }  
}

**55) How to set the custom action filter in action methods in Web API?**

Below is the sample code of action with custom action filter –

public class MyCustomerTestController : ApiController  
{  
 [MyCustomModelAttribute]  
 public HttpResponseMessage Post(MyTestCustomer customer)  
 {  
 // ...  
 }  
}

**56) What is BSON in Web API?**

It’s is a binary serialization format. “BSON” stands for “Binary JSON”. BSON serializes objects to key-value pair as in JSON. Its light weight and its fast in encode/decode.

**57) How to enable BSON in server?**

Add “BsonMediaTypeFormatter” in WebAPI.config as shown below

public static class WebApiConfig  
{  
 public static void Register(HttpConfiguration config)  
 {  
 config.Formatters.Add(new BsonMediaTypeFormatter());  
 // Other Web API configuration goes here  
 }  
}

**58) How parameter binding works in Web API?**

Below are the rules followed by WebAPI before binding parameters –

* If it is simple parameters like – bool,int, double etc. then value will be obtained from the URL.
* Value read from message body in case of complex types.

**59) Why to use “FromUri” in Web API?**

In Web API to read complex types from URL we will use “FromUri” attribute to the parameter in action method. Eg:

public MyValuesController : ApiController  
{  
 public HttpResponseMessage Get([FromUri] MyCustomer c) { ... }  
}

**60) Why to use “FromBody” in Web API?**

This attribute is used to force Web API to read the simple type from message body. “FromBody” attribute is along with parameter. Eg:

public HttpResponseMessage Post([FromBody] int customerid, [FromBody] string customername) { ... }

**61) Why to use “IValueprovider” interface in Web API?**

This interface is used to implement custom value provider.

**1) What is Web API?**

WebAPI is a framework which helps you to build/develop HTTP services.

**2) Why is Web API required? Is it possible to use RESTful services using WCF?**

Yes, we can still develop RESTful services with WCF. However, there are two main reasons that prompt users to use Web API instead of RESTful services.

* Web API increases TDD (Test Data Driven) approach in the development of RESTful services.
* If we want to develop RESTful services in WCF, you surely need a lot of config settings, URI templates, contracts & endpoints for developing RESTful services using web API.

**3) Why select Web API?**

* It is used to create simple, non-SOAP-based HTTP Services
* It is also an easy method for creation with Web API. With WCF REST Services
* It is based on HTTP and easy to define, expose and consume in a REST-ful way.
* It is lightweight architecture and ideal for devices that have limited bandwidth like smartphones.

**4) Is it right that ASP.NET Web API has replaced WCF?**

It’s a not at all true that ASP.NET Web API has replaced WCF. In fact, it is another way of building non-SOAP based services, i.e., plain XML or JSON string.

**5) What are the advantages of Web API?**

Advantages of Web API are:

* OData
* Filters
* Content Negotiation
* Self-Hosting
* Routing
* Model Bindings

[](https://cdn.career.guru99.com/wp-content/uploads/2017/11/webapi-odata-13.png)

**6) What are main return types supported in Web API?**

A Web API controller action can return following values:

* Void – It will return empty content
* HttpResponseMessage – It will convert the response to an HTTP message.
* IHttpActionResult – internally calls ExecuteAsync to create an HttpResponseMessage
* Other types – You can write the serialized return value into the response body

**7) Web API supports which protocol?**

Web App supports HTTP protocol.

**8) Which .NET framework supports Web API?**  
NET 4.0 and above version supports web API.

**9) Web API uses which of the following open-source library for JSON serialization?**

Web API uses Json.NET library for JSON serialization.

**10) By default, Web API sends HTTP response with which of the following status code for all uncaught exception?**

500 – Internal Server Error

**11) What is the biggest disadvantage of “Other Return Types” in Web API?**

The biggest disadvantage of this approach is that you cannot directly return an error code like 404 error.

**12) How do you construct HtmlResponseMessage?**

Following is the way to construct to do so,



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | public class TestController : ApiController    {    public HttpResponseMessage Get()    {    HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.OK, "value");    response.Content = new StringContent("Testing", Encoding.Unicode);    response.Headers.CacheControl = new CacheControlHeaderValue()    {    MaxAge = TimeSpan.FromMinutes(20)    };    return response;    }    } |

**13) What is Web API Routing?**

Routing is pattern matching like in MVC.

All routes are registered in Route Tables.

For example:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | Routes.MapHttpRoute(    Name: "ExampleWebAPIRoute",    routeTemplate: “api/{controller}/{id}    defaults: new { id = RouteParameter.Optional} |

**14) What is SOAP?**

SOAP is an XML message format used in web service interactions. It allows to send messages over HTTP or JMS, but other transport protocols can be used. It is also an XML-based messaging protocol for exchanging information among computers.

**15) What is the benefit of using REST in Web API?**

REST is used to make fewer data transfers between client and server which make it an ideal for using it in mobile apps. Web API also supports HTTP protocol. Therefore, it reintroduces the traditional way of the HTTP verbs for communication.

**16) How can we use Web API with ASP.NET Web Form?**

Web API can be used with ASP.NET Web Form

It can be performed in three simple steps:

1. Create a Web API Controller,
2. Add a routing table to Application\_Start method of Global.sax
3. Then you need to make a jQuery AJAX Call to Web API method and get data.

**17) How to you can limit Access to Web API to Specific HTTP Verb?**

Attribute programming plays a important role. It is easy to restrict access to an ASP.NET Web API method to be called using a particular HTTP method.

**18) Can you use Web API with ASP.NET Web Form?**

Yes, It is possible to use Web API with ASP.Net web form. As it is bundled with ASP.NET MVC framework. However, it can be used with ASP.NET Web Form.

**19) How Can assign alias name for ASP.NET Web API Action?**

We can give alias name for Web API action same as in case of ASP.NET MVC by using “ActionName” attribute as follows:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | [HttpPost]    [ActionName("SaveStudentInfo")]    public void UpdateStudent(Student aStudent)  {  StudentRepository.AddStudent(aStudent);  } |

**20) What is the meaning of TestApi?**

TestApi is a utility library of APIs. Using this library tester developer can create testing tools and automated tests for a .NET application using data-structure and algorithms.

**21) Explain exception filters?**

It will be executed when exceptions are unhandled and thrown from a controller meth*o*d. The reason for the exception can be anything. Exception filters will implement “IExceptionFilter” interface.

**22) How can we register exception filter from the action?**

We can register exception filter from action using following code:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | [NotImplExceptionFilter]    public TestCustomer GetMyTestCustomer(int custid)    {    //write the code    } |

**23) How you can return View from ASP.NET Web API method?**

No, we can’t return a view from ASP.NET Web API Method. Web API creates HTTP services that render raw data. However, it’s also possible in ASP.NET MVC application.

**24) How to register exception filter globally?**

It is possible to register exception filter globally using following code-

GlobalConfiguration.Configuration.Filters.Add(new MyTestCustomerStore.NotImplExceptionFilterAttribute());

**25) Explain what is REST and RESTFUL?**

REST represents REpresentational  State Transfer; it is entirely a new aspect of writing a web app.

RESTFUL: It is term written by applying REST architectural concepts is called RESTful services. It focuses on system resources and how the state of the resource should be transported over HTTP protocol.

**26) Give me one example of Web API Routing?**



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | Config.Routes.MapHttpRoute(    name: "MyRoute,"//route name    routeTemplate: "api/{controller}/{action}/{id}",//as you can see "API" is at the beginning.    defaults: new { id = RouteParameter.Optional }    ); |

**27) How can you handle errors in Web API?**

Several classes are available in Web API to handle errors. They are HttpError, Exception Filters, HttpResponseException, and Registering Exception Filters.

**28) What New Features comes with ASP.NET Web API 2.0?**

The latest features of ASP.NET Web API framework v2.0 are as follows:

* Attribute Routing
* Cross-Origin Resource Sharing
* External Authentication
* Open Web Interface NET
* HttpActionResult
* Web API OData

**29) How can you restrict access methods to specific HTTP verbs in Web API?**

With the help of Attributes (like HTTP verbs), It is possible to implement access restrictions in Web API.

It is possible to define HTTP verbs as an attribute to restrict access.

Example:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | [HttpPost]    public void Method1(Class obj)    {    //logic |

**30) How can you pass multiple complex types in Web API?**

Two methods to pass the complex types in Web API –

Using ArrayList and Newtonsoft array

**31) Write a code for passing ArrayList in Web API?**



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | ArrayList paramList = new ArrayList();    Category c = new Category { CategoryId = 1, CategoryName =“MobilePhones”};    Product p = new Product { Productcode = 1, Name = “MotoG”, Price = 15500, CategoryID = 1 };    paramList.Add(c);    paramList.Add(p); |

**32) Name the tools or API for developing or testing web api?**

Testing tools for web services for REST APIs include:

1. Jersey API
2. CFX
3. Axis
4. Restlet

**33) What is REST?**

REST is architectural style. It has defined guidelines for creating services which are scalable. REST used with HTTP protocol using its verbs GET, PUT, POST and DELETE.

**34) How to unit test Web API?**

We can perform a Unit test using Web API tools like Fiddler.

Here, are some setting to be done if you are using

Fiddler –Compose Tab -> Enter Request Headers -> Enter the Request Body and execute

**35) How can we restrict access to methods with specific HTTP verbs in Web API?**

Attribute programming is widely used for this functionality. Web API also allows restricting access of calling methods with the help of specific HTTP verbs. It is also possible to define HTTP verbs as attribute over method.

**36) What is the usage of DelegatingHandler?**

DelegatingHandler is used in the Web API to represent Message Handlers before routing.

**37) How can we register exception filter from the action?**

We can register exception filter from action using following code



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | [NotImplExceptionFilter]    public TestCust GetMyTestCust (int custno)    {    //write the code    } |

**38) Tell me the code snippet to show how we can return 404 errors from HttpError?**

Code for returning 404 error from HttpError

string message = string.Format(“TestCustomer id = {0} not found”, customerid);

return Request.CreateErrorResponse(HttpStatusCode.NotFound, message);

**39) Explain code snippet to register exception filters from controller?**



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | [NotImplExceptionFilter]    public class TestCustController : Controller    {    //Your code goes here    } |

**40) Web API supports which protocol?**

Web App support HTTP protocol

**41) Which of the following .NET framework supports Web API?**

Web API is supported by NET 4.0 version

**42) Web API uses which library for JSON serialization?**

Web API uses Json.NET library for JSON serialization.

**43) By default, Web API sends HTTP response with which of the following status code for all uncaught exception?**

500 – Internal Server Error

**44) Explain method to handle error using HttpError in Web API?**

In WEB API HttpError used to throw the error info in the response body. “CreateErrorResponse” method is can also use along with this, which is an extension method defined in “HttpRequestMessageExtension.”

**45) How can we register exception filter globally?**

We can register exception filter globally using following code:



|  |  |
| --- | --- |
| 1 | GlobalConfiguration.Configuration.Filters.Add (new MyTestCustomerStore.NotImplExceptionFilterAttribute()); |

**46) How to handle errors in Web API?**

Several classes are available in Web API to handle errors. They are HttpError, HttpResponseException, Exception Filters, Registering Exception Filters.

**47) What is the benefit of WebAPI over WCF?**

WCF services use the SOAP protocol while HTTP never use SOAP protocol. That’s why WebAPI services are lightweight since SOAP is not used. It also reduces the data which is transferred to resume service. Moreover, it never needs too much configuration. Therefore, the client can interact with the service by using the HTTP verbs.

**48) State differences between MVC and WebAPI**

MVC framework is used for developing applications which have User Interface. For that, views can be used for building a user interface.

WebAPI is used for developing HTTP services. Other apps can also be called the WebAPI methods to fetch that data.

**49) Who can consume WebAPI?**

WebAPI can be consumed by any client which supports HTTP verbs such as GET, PUT, DELETE, POST. As WebAPI services don’t need any configuration, they are very easy to consume by any client. Infract, even portable devices like Mobile devices can easily consume WebAPI which is certainly the biggest advantages of this technology.

**50) How can we make sure that Web API returns JSON data only?**

To make Web API serialize the returning object to JSON format and returns JSON data only. For that you should add the following code in WebApiConfig.cs class in any MVC Web API Project:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | //JsonFormatter    //MediaTypeHeaderValue    Config.Formatters.JsonFormatter.SupportedMediaTypes.Add(new MediaTypeHeaderValue("application/json"));    1    2    3    //JsonFormatter    //MediaTypeHeaderValue    Config.Formatters.JsonFormatter.SupportedMediaTypes.Add(new MediaTypeHeaderValue("application/json")) |

