WEB API

1)Advantages :

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

2)Return Types supported in Web Api?

* 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

3) HTTP Response Status code

* Informational responses ( 100 – 199 )
* Successful responses ( 200 – 299 )
* Redirection messages ( 300 – 399 )
* Client error responses ( 400 – 499 )
* Server error responses ( 500 – 599 )

4) Web Api Routing

Routes.MapHttpRoute(

Name: "ExampleWebAPIRoute",

routeTemplate: “api/{controller}/{id}

defaults: new { id = RouteParameter.Optional}

5) Alias Name for web api method?

[HttpPost]

[ActionName("SaveStudentInfo")]

public void UpdateStudent(Student aStudent)

{

StudentRepository.AddStudent(aStudent);

}

6) Code for returning 404 error from HttpError

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

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

7) register exception filter in webapi?

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

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

9)

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.

10)

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.

11) **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:

//JsonFormatter

//MediaTypeHeaderValue

Config.Formatters.JsonFormatter.SupportedMediaTypes.Add(new MediaTypeHeaderValue("application/json"))

12)

* It is used to provide an interface for websites and client applications to have access to data.
* It can also be used to access data from the database and save data back to the database.
* It supports different text formats such as XML, JSON, etc.
* It is suitable or compatible with any type of browser and any type of device like mobile, desktop, web, etc.
* It uses low bandwidth such as XML or JSON data, etc., and is therefore considered good for devices that have limited bandwidth such as smartphones, etc.
* From a business point of view, web API is more applicable for UI/UX, increases interest in the company’s product and services, increases website traffic.

13)

Types of Media type formatters

|  |  |  |
| --- | --- | --- |
| JsonMediaTypeFormatter | application/json, text/json | Handles JSON format |
| XmlMediaTypeFormatter | application/xml, text/json | Handles XML format |
| FormUrlEncodedMediaTypeFormatter | application/x-www-form-urlencoded | Handles HTM form URL-encoded data |
| JQueryMvcFormUrlEncodedFormatter | application/x-www-form-urlencoded | Handles model-bound HTML form URL-encoded data |

14) Web Api Filters

* **Authentication Filter:** It handles authentication and authenticates HTTP requests. It also helps to authenticate user detail. It checks the identity of the user.
* **Authorization Filter:** It handles authorization. It runs before controller action. This filter is used to check whether or not a user is authenticated. If the user is not authenticated, then it returns an HTTP status code 401 without invoking the action.
* **AuthorizeAttribute** is a built-in authorization filter provided by Web API.
* **Action Filter**: It is attributing that one can apply to controller action or entire controller. It is used to add extra logic before or after controller action executes. It is simply a way to add extra functionality to Web API services.
* **Exception Filter**: It is used to handle exceptions that are unhandled in Web API. It is used whenever controller actions throw an unhandled exception that is not HttpResponseException. It will implement an “IExceptionFilter” interface.
* **Override Filter**: It is used to exclude specific action methods or controllers from the global filter or controller level filter. It is simply used to modify the behaviour of other filters for individual action methods.

15)

HttpConfiguration properties in Webapi

* **DependencyResolver:** It sets or gets a dependency resolver for dependency injection.
* **Services:**It gets web API services.
* **ParameterBindingRules:** It gets a collection of rules for how parameters should be bound.
* **MessageHandlers:**  It sets or gets message handlers.
* **Formatters:** It sets or gets media-type formatters.

16)Content Negotiation

Content negotiation is basically a process of selecting the best representation from multiple representations that are available for a given response. It simply allows one to choose rather than negotiate content that one wants to get in response. It is performed at the server-side. In simple words, it chooses the best media type for matters to return a response to an incoming request.

17) CORS(cross Origin resource sharing)

We can enable CORS in WebAPI,

1. Using JSONP
2. var jsonpFormatter = **new** JsonpMediaTypeFormatter(config.Formatters.JsonFormatter);
3. config.Formatters.Add(jsonpFormatter);

2.Using Microsoft.AspNet.WebApi.Cors

1. [EnableCors(origins: "\*", headers: "\*", methods: "\*", exposedHeaders: "X-My-Header")]
2. **public** **class** TestController : ApiController
3. {
4. }

17) Delegating Handlers

DelegatingHandler is used to develop a custom Server-Side HTTP Message Handler in ASP.NET Web API. It is used to represent Message Handlers before routing in Web API.