1. What is Web API - : ApiController

Web API is framework that help us **to build RESTFUL service or HTTP based service**.

Web API can be consumed from **broad range of client**

1. Browsers  
2. Mobile applications  
3. Desktop applications  
4. IOTs

1. What are RESTful services

REST stands for Representational State Transfer.

REST is an architectural pattern for creating an API that uses HTTP as its underlying communication method

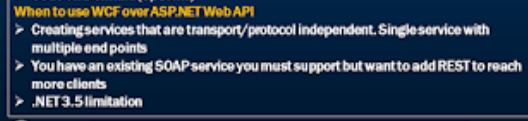
REST architectural pattern specifies a set of constraints that a system should adhere to

Resource is and the HTTP verbs - GET, PUT, POST & DELETE

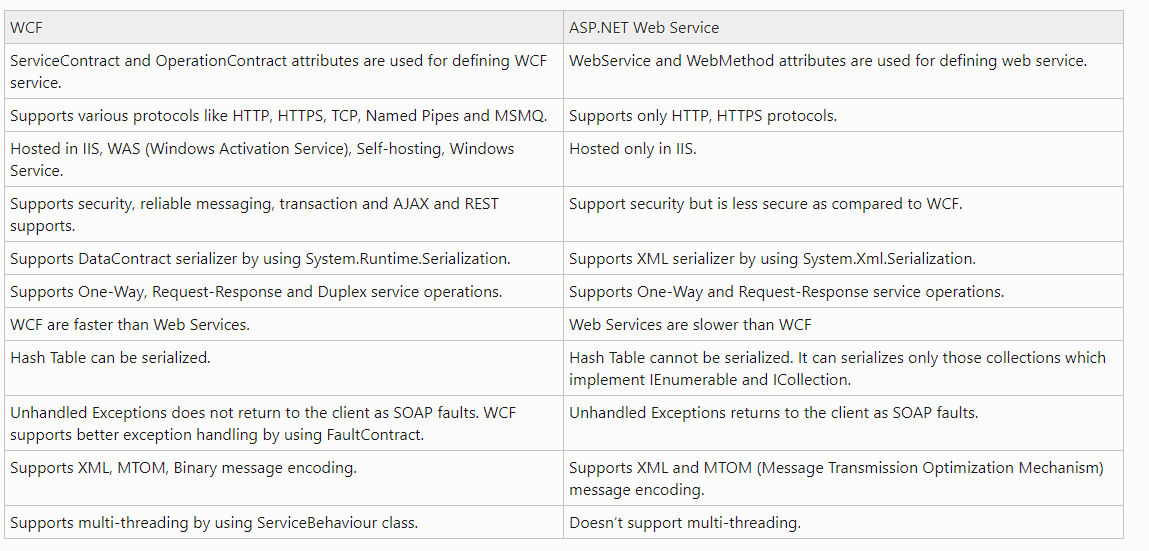
* Resources typically represent data entities. **Product, Employee**, Customer etc are all resources
* The HTTP verb (GET, PUT, POST, DELETE) that is sent with each request tells the API what to do with the resource

| **Resource** | **Verb** | **Outcome** |
| --- | --- | --- |
| /Employees | GET | Gets list of employees |
| /Employee/1 | GET | Gets employee with Id = 1 |
| /Employees | POST | Creates a new employee |
| /Employee/1 | PUT | Updates employee with Id = 1 |
| /Employee/1 | DELETE | Deletes employee with Id = 1 |

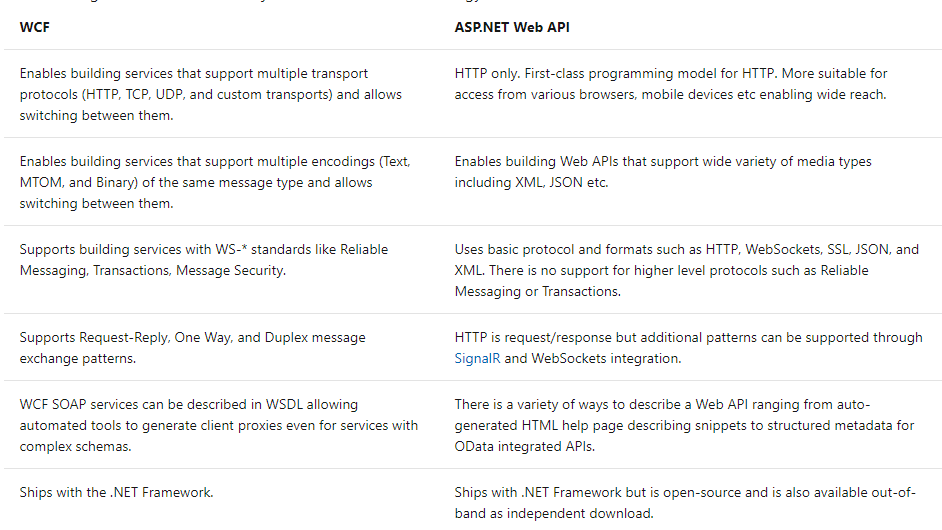
1. When to use WCF over WEB API



1. Difference between WCF and ASP.NET Web Service



1. WCF vs WEB API



1. Web API Content Negotiation

* the client have ability to decide in which format they want the response - XML, JSON etc
* A request that is sent to the server includes an Accept header. Using the Accept header the client can specify the format for the response. For example  
    
  Accept: application/xml returns XML  
  Accept: application/json returns JSON
* Depending on the Accept header value in the request, the server sends the response. This is called Content Negotiation
* **XML formatter or Json formatter used** by web API based on client request header

1. Media type formatters

* XML formatter or Json formatter used by web API **based on client request header** is called media type formatter
* If client don't specify the Accept header, by default the Web API returns JSON data.
* Our own formatters, for custom formatting the data
* quality factor (for XML or Json) - higher quality factor win and according to higher quality factor formatter is used

1. Content-Type header and Accept header s

* The formatters are used by the server for both request and response messages
* When the client sends a request to the server, we set the **Content-Type header** to the appropriate value to let the server know the format of the data that we are sending
  + example, if the client is sending JSON data, the Content-Type header is set to application/json. The server knows it is dealing with JSON data, so it uses JSON formatter to convert JSON data to .NET Type
* Similarly when a response is being sent from the server to the client, depending on the Accept header value, the appropriate formatter is used to convert .NET type to JSON, XML etc.
* very easy to change the serialization settings of these formatters, WebApiConfig.cs file in App\_Start folder.
* Content-Type tell Web API at server end in which formate data is coming
* Accept header tell web api that in which format I want data back

1. 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 irrespective of the Accept header value**

WebApiConfig.cs file in App\_Start folder

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

* **How to return only XML of the Accept header value**

WebApiConfig.cs file in App\_Start folder

config.Formatters.Remove(config.Formatters.JsonFormatter);

1. FromBody and FromUri in Web API

* **Web API for binding parameters**
  + if the parameter is a simple type like int, bool, double, etc.,
    - Web API tries to get the value from the URI (Either from route data or Query String)
  + If the parameter is a complex type like Customer, Employee etc.,
    - Web API tries to get the value from the request body
* We can change this default **parameter binding** process by using [FromBody] and [FromUri] attributes.

public HttpResponseMessage Put([FromBody]int id, [FromUri]Employee employee)

{}

1. Versioning

* **Different options available to version Web API services :**
* Versioning can be implemented using  
  1. URI's  
  2. Query String  
  3. Version Header  
  4. Accept Header  
  5. Media Type