WCF:

WCF is the latest programming model for building and developing service-oriented application. WCF allows applications to communicate with each other in distributed environment. WCF is a set of technologies that covers ASMX web services, Web Services Enhancements (WSE), .NET Remoting and MSMQ. It enables developers to build secure, reliable, transacted solutions that integrate across platforms and interoperable with existing.

WCF is an umbrella technology that covers ASMX web services, .NET remoting, WSE, Enterprise Service, and System.Messaging. It is designed to offer a manageable approach to distributed computing, broad interoperability, and direct support for service orientation. WCF supports many styles of distributed application development by providing a layered architecture.

ABC of Windows Communication Foundation

What are the ABCs of WCF? "ABC" stands for address, binding and contract.

Address (Where)

It specifies the location of the service means, where the service is hosted. The service hosting URL may be like http://server/wcfService. Clients will use this location to communicate with your service.

Binding (How)

It specifies how the client will communicate to the service. We have different protocols (like http,tcp,named pipe,msmq) for the WCF to communicate to the client.

Contract (What)

It specifies what the service will do. For this purpose we have different contract like as Data Contract, Operation Contract, Message Contract, Fault Contract. I will discuss all these later.

WCF Hosting

A WCF service is a component that can be called by other applications. It must be hosted in an environment in order to be discovered and used by others. The WCF host is an application that controls the lifetime of the service. With .NET 3.0 and beyond, there are several ways to host the service.

Self hosting

A WCF service can be self-hosted, which means that the service runs as a standalone application and controls its own lifetime. This is the most flexible and easiest way of hosting a WCF service, but its availability and features are limited.

Windows services hosting

A WCF service can also be hosted as a Windows service. A Windows service is a process managed by the operating system and it is automatically started when Windows is started (if it is configured to do so). However, it lacks some critical features (such as versioning) for WCF services.

IIS hosting

A better way of hosting a WCF service is to use IIS. This is the traditional way of hosting a web service. IIS, by nature, has many useful features, such as process recycling, idle shutdown, process health monitoring, message-based activation, high availability, easy manageability, versioning, and deployment scenarios. All of these features are required for enterprise-level WCF services.

Windows Activation Services hosting

The IIS hosting method, however, comes with several limitations in the service-orientation world; the dependency on HTTP is the main culprit. With IIS hosting, many of WCF's flexible options can't be utilized. This is the reason why Microsoft specifically developed a new method, called Windows Activation Services, to host WCF services.

Windows Process Activation Service (WAS) is the new process activation mechanism for Windows Server 2008 that is also available on Windows Vista. WAS hosting is possible only with IIS 7.0.Additional WCF components also plug into WAS to provide message-based activation over the other protocols that WCF supports, such as TCP, MSMQ, and named pipes. This allows applications that use the non-HTTP communication protocols to use the IIS features such as process recycling, rapid fail protection, and the common configuration systems that were only available to HTTP-based applications.

Difference between WCF and ASP.NET Web Service

WCF	ASP.NET Web Service
ServiceContract and OperationContract attributes are used for defining WCF service.	WebService and WebMethod attributes are used for defining web service.
Supports various protocols like HTTP, HTTPS, TCP, Named Pipes and MSMQ.	Supports only HTTP, HTTPS protocols.
Hosted in IIS, WAS (Windows Activation Service), Self-hosting, Windows Service.	Hosted only in IIS.
Supports security, reliable messaging, transaction and AJAX and REST supports.	Support security but is less secure as compared to WCF.
Supports DataContract serializer by using System.Runtime.Serialization.	Supports XML serializer by using System.Xml.Serialization.
Supports One-Way, Request-Response and Duplex service operations.	Supports One-Way and Request-Response service operations.
WCF are faster than Web Services.	Web Services are slower than WCF
Hash Table can be serialized.	Hash Table cannot be serialized. It can serializes only those collections which implement IEnumerable and ICollection.
Unhandled Exceptions does not return to the client as SOAP faults. WCF supports better exception handling by using FaultContract.	Unhandled Exceptions returns to the client as SOAP faults.
Supports XML, MTOM, Binary message encoding.	Supports XML and MTOM (Message Transmission Optimization Mechanism) message encoding.
Supports multi-threading by using ServiceBehaviour class.	Doesn’t support multi-threading.

WCF Version History

WCF Version	Introduced with .NET & IDE	Features Detail
4.5	4.5 and Visual Studio 2012	1. Task-based Async Support 2. Contract-First Development 3. WCF Configuration Validation 4. Web Socket Support 5. UDP Endpoint Support 6. New Https protocol mapping on IIS 7. Streaming Improvements 8. Multiple Auth support for single endpoint
4.0	4.0 and Visual Studio 2010	1. Simple Configuration 2. Serialization Enhancements 3. Support for WS-Discovery 4. Routing Service 5. Standard Endpoints 6. Workflow Service
3.5	3.5 and Visual Studio 2008	1. Uri Templates Support 2. Support for REST Style Services 3. Asp.NET Ajax Integration and JSON support 4. Added WS* Specification Support 5. Support for RSS and Atom feeds
3.0	3.0 and Visual Studio 2005	1. Introduced first version of WCF with many features like Address, Binding, Contract, Sessions, Instancing, and Concurrency management

Understanding various types of WCF bindings

WCF binding is a set of binding elements and each element specify, how the service and client will communicates with each other's. Each binding must have at least one transport element and one message encoding element.

Different types of WCF bindings

WCF has a couple of built in bindings which are designed to fulfill some specific need. You can also define your own custom binding in WCF to fulfill your need. All built in bindings are defined in the System.ServiceModel Namespace. Here is the list of 10 built in bindings in WCF which we commonly used:

Basic binding

This binding is provided by the BasicHttpBinding class. It is designed to expose a WCF service as an ASMX web service, so that old clients (which are still using ASMX web service) can consume new service. By default, it uses Http protocol for transport and encodes the message in UTF - 8 text for-mat. You can also use Https with this binding.

Web binding

This binding is provided by the WebHttpBinding class. It is designed to expose WCF services as Http requests by using HTTP-GET, HTTP-POST. It is used with REST based services which may give output as an XML or JSON format. This is very much used with social networks for implementing a syndication feed.

Web Service (WS) binding

This binding is provided by the WSHttpBinding class. It is like as Basic binding and uses Http or Https protocols for transport. But this is designed to offer various WS - * specifications such as WS – Reliable Messaging, WS - Transactions, WS - Security and so on which are not supported by Basic binding.

> wsHttpBinding= basicHttpBinding + WS-* specification

WS Dual binding

This binding is provided by the WsDualHttpBinding class. It is like as wsHttpBinding except it sup-ports bi-directional communication means both clients and services can send and receive messages.

TCP binding

This binding is provided by the NetTcpBinding class. It uses TCP protocol for communication be-tween two machines with in intranet (means same network). It encodes the message in binary format. This is faster and more reliable binding as compared to the Http protocol bindings. It is only used when communication is WCF - to – WCF means both client and service should have WCF.

IPC binding

This binding is provided by the NetNamedPipeBinding class. It uses named pipe for Communication between two services on the same machine. This is the most secure and fastest binding among all the bindings.

MSMQ binding

This binding is provided by the NetMsmqBinding class. It uses MSMQ for transport and offers sup-port to disconnected message queued. It provides solutions for disconnected scenarios in which service processes the message at a different time than the client send the messages.

Federated WS binding

This binding is provided by the WSFederationHttpBinding class. It is a specialized form of WS binding and provides support to federated security.

Peer Network binding

This binding is provided by the NetPeerTcpBinding class. It uses TCP protocol but uses peer net-working as transport. In this networking each machine (node) acts as a client and a server to the other nodes. This is used in the file sharing systems like torrent.

MSMQ Integration binding

This binding is provided by the MsmqIntegrationBinding class. This binding offers support to communicate with existing systems that communicate via MSMQ.

Understanding different types of WCF Contracts

WCF contract specify the service and its operations. WCF has five types of contracts: service contract, operation contract, data contract, message contract and fault contract.

Service Contract

A service contract defines the operations which are exposed by the service to the outside world. A service contract is the interface of the WCF service and it tells the outside world what the service can do. It may have service-level settings, such as the name of the service and namespace for the service.

```
[ServiceContract]
interface IMyContract
{
[OperationContract]
string MyMethod();
}

class MyService : IMyContract
{
public string MyMethod()
{
return "Hello World";
}
}
```

Operation Contract

An operation contract is defined within a service contract. It defines the parameters and return type of an operation. An operation contract can also defines operation-level settings, like as the transaction flow of the op-eration, the directions of the operation (one-way, two-way, or both ways), and fault contract of the operation.

```
[ServiceContract]
interface IMyContract
{
[FaultContract(typeof(MyFaultContract))]
[OperationContract]
string MyMethod();
}
```

Data Contract

A data contract defines the data type of the information that will be exchange be-tween the client and the service. A data contract can be used by an operation contract as a parameter or return type, or it can be used by a message contract to define elements.

```
[DataContract]
class Person
{
[DataMember]
public string ID;
[DataMember]
public string Name;
}

[ServiceContract]
interface IMyContract
{
[OperationContract]
Person GetPerson(int ID);
}
```

Message Contract

When an operation contract required to pass a message as a parameter or return value as a message, the type of this message will be defined as message contract. A message contract defines the elements of the message (like as Message Header, Message Body), as well as the message-related settings, such as the level of message security.

Message contracts give you complete control over the content of the SOAP header, as well as the structure of the SOAP body.

```
[ServiceContract]
public interface IRentalService
{
[OperationContract]
double CalPrice(PriceCalculate request);
}

[MessageContract]
public class PriceCalculate
{
[MessageHeader]
public MyHeader SoapHeader { get; set; }
[MessageBodyMember]
public PriceCal PriceCalculation { get; set; }
}

[DataContract]
public class MyHeader
{
[DataMember]
public string UserID { get; set; }
}

[DataContract]
public class PriceCal
{
[DataMember]
public DateTime PickupDateTime { get; set; }
[DataMember]
public DateTime ReturnDateTime { get; set; }
[DataMember]
public string PickupLocation { get; set; }
[DataMember]
public string ReturnLocation { get; set; }
}
```

Fault Contract

A fault contract defines errors raised by the service, and how the service handles and propagates errors to its clients. An operation contract can have zero or more fault contracts associated with it.

```
[ServiceContract]
interface IMyContract
{
[FaultContract(typeof(MyFaultContract1))]
[FaultContract(typeof(MyFaultContract2))]
[OperationContract]
string MyMethod();

[OperationContract]
string MyShow();
}
```

Interview Questions in WCF

What is WCF?

Microsoft refers WCF as a programming platform that is used to build Service-oriented applications. Windows Communication Foundation is basically a unified programming model for developing, configuring and deploying distributed services. Microsoft has unified all its existing distributed application technologies (e.g. MS Enterprise Services, ASMX web services, MSMQ, .NET Remoting etc) at one platform i.e. WCF. Code name for WCF was Indigo.

Why to use WCF? or What are the advantages for using WCF?

- Service Orientation is one of the key advantages of WCF. We can easily build service-oriented applications using WCF.
- If compared with ASMX web services, WCF service provides reliability and security with simplicity.
- As oppose to .NET Remoting, WCF services are interoperable.
- Different clients can interact with same service using different communication mechanism. This is achieved by using service endpoints. A single WCF service can have multiple endpoints. So, developer will write code for service once and just by changing configuration (defining another service endpoint), it will be available for other clients as well.
- Extensibility is another key advantage of WCF. We can easily customize a service behavior if required.

What are the core components of WCF Service?

A WCF service has at least following core components.

- Service Class: A ervice class implementing in any CLR-based language and expose at least one method.
- Hosting Environment: a managed process for running service.
- Endpoint: a client uses it to communicate with service.

What is the difference between WCF and ASMX Web services?

The basic difference is that ASMX web service is designed to send and receive messages using SOAP over HTTP only. While WCF service can exchange messages using any format (SOAP is default) over any transport protocol (HTTP, TCP/IP, MSMQ, Named Pipes etc).

You can find detailed discussion on WCF Vs ASMX Web services here.

What are the Endpoints in WCF? or Explain ABCs of endpoint?

For WCF services to be consumed, it's necessary that it must be exposed; Clients need information about service to communicate with it. This is where service endpoints play their role.

A service endpoint has three basic elements or also called ABCs of an endpoint i.e. Address, Binding and Contract.

Address: It defines "WHERE". Address is the URL that identifies the location of the service.

Binding: It defines "HOW". Binding defines how the service can be accessed.

Contract: It defines "WHAT". Contract identifies what is exposed by the service.

What is a WCF Binding? How many different types of bindings available in WCF?

Bindings in WCF actually defines that how to communicate with the service. Binding specifies that what communication protocol as well as encoding method will be used. Optionally, binding can specify other important factors like transactions, reliable sessions and security.

Another WCF Tutorial gives more detailed understanding of Binding concept in WCF.

There are different built-in bindings available in WCF, each designed to fulfill some specific need.

- basicHttpBinding
- wsHttpBinding
- netNamedPipeBinding
- netTcpBinding
- netPeerTcpBinding
- netmsmqBinding

For details on different binding types, please follow the link to WCF bindings.

Can we have multiple endpoints for different binding types in order to serve different types of clients?

Yes, we can have multiple endpoints for different binding types. For example, an endpoint with wsHttpBinding and another one with netTcpBinging.

What are the hosting options for WCF Services? Explain.

For a service to host, we need at least a managed process, a ServiceHost instance and an Endpoint configured. Possible approaches for hosting a service are:

1. Hosting in a Managed Application/ Self Hosting

a. Console Application

b. Windows Application

c. Windows Service

2. Hosting on Web Server

a. IIS 6.0 (ASP.NET Application supports only HTTP)

b. Windows Process Activation Service (WAS) i.e. IIS 7.0 supports HTTP, TCP, NamedPipes, MSMQ.

What are Contracts in WCF?

A Contract is basically an agreement between the two parties i.e. Service and Client. In WCF, Contracts can be categorized as behavioral or structural.

1 *Behavioral Contracts* define that what operations client can perform on a service.

- *ServiceContract* attribute is used to mark a type as Service contract that contains operations.
- *OperationContract* attributes is used to mark the operations that will be exposed.
- *Fault Contract* defines what errors are raised by the service being exposed.
- *Structural ContractsDataContract* attribute define types that will be moved between the parties.
- *MessageContract* attribute define the structure of SOAP message.

What Message Exchange Patterns supported by WCF?

- Request/Response
- One Way
- Duplex

Request/Response

It's the default pattern. In this pattern, a response message will always be generated to consumer when the operation is called, even with the void return type. In this scenario, response will have empty SOAP body.

One Way

In some cases, we are interested to send a message to service in order to execute certain business functionality but not interested in receiving anything back.

OneWay MEP will work in such scenarios.If we want queued message delivery, OneWay is the only available option.

Duplex

The Duplex MEP is basically a two-way message channel. In some cases, we want to send a message to service to initiate some longer-running processing and require a notification back from service in order to confirm that the requested process has been completed.

Next ***WCF Tutorial on Interview Questions and Answers*** in this series is about Proxy and Channel Factory, Concurrency and Throttling in WCF.

What is a Service Proxy in Windows Communication Foundation?

A service proxy or simply proxy in WCF enables application(s) to interact with WCF Service by sending and receiving messages. It's basically a class that encapsulates service details i.e. service path, service implementation technology, platform and communication protocol etc. It contains all the methods of service contract (signature only, not the implementation). So, when the application interact the service through proxy, it gives the impression that it's communicating a local object.

We can create proxy for a service by using Visual Studio or SvcUtil.exe.

What is WCF throttling?

WCF throttling enables us to regulate the maximum number of WCF instances, concurrent calls and concurrent sessions. Basic purpose is to control our WCF service performance by using Service throttling behavior.

In configuration file we can set this behavior as follows:

```
<serviceBehavior>
<behavior name=”MyServiceBehavior”>
<serviceThrottling
maxConcurrentInstances=”2147483647”
maxConcurrentCalls=”16″
maxConcurrentSessions=”10″
</behavior>
</serviceBehavior>
```

Above given values are the default ones, but we can update it after looking into the requirements of our application.

What is a fault contract?

Normally, by default, when some exception occurs at a WCF service level, it will not expose as it is to client. Reason is that WCF exception is a CLR exception and it doesn't make sense to expose it outside of CLR because it contains internal details of service code like stack trace. So, WCF handles and returns error details to client using Fault Contract."So, fault contract is a contract that contains the details of possible exception(s) that might occur in a service code."

```
[ServiceContract]
public interface IService1
{
[OperationContract]
[FaultContract(typeof(MyFaultDetails))]
int MyOperation1();
}

[DataContract]
public class MyFaultDetails
{
[DataMember]
public string ErrorDetails { get; set; }
}
```

In implementing service…..

```
public int MyOperation1()
{
Try{

//Do something……

}catch()
{
MyFaultDetails ex = new MyFaultDetails();
ex.ErrorDetails = "Specific error details here.";
throw new FaultException(ex,"Reason: Testing…..");
}
}
```

A user has a service with a one-way operation that includes a fault contract, and he gets an exception when he tries to host the service. Why?

This is true, because, to return faults, the service requires some form of a two-way communication channel, which is not there with one-way operations.

Different Versions of MVC

ASP.NET MVC 3

- New Project Templates having support for **HTML 5** and **CSS 3**.
- Improved Model validation.
- Razor View Engine introduced with a bundle of new features.
- Having support for Multiple View Engines i.e. *Web Forms* view engine, *Razor* or open source.
- Controller improvements like *ViewBag* property and *ActionResults* Types etc.
- Unobtrusive *JavaScript* approach.
- Improved Dependency Injection with new *IDependencyResolver.*
- Partial page output caching.

ASP.NET MVC 4

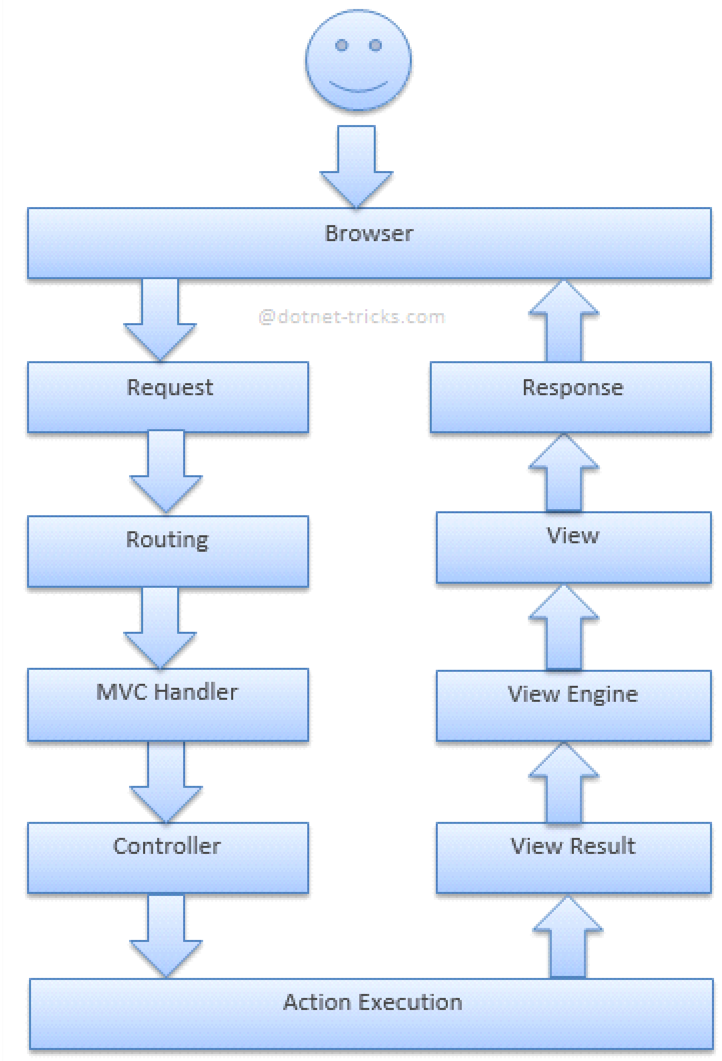
- *ASP.NET Web API*, a framework that simplifies the creation of *HTTP services* and serving a wide range of clients. Follow to create your first ASP.NET Web API service.
- Adaptive rendering and other look-n-feel improvements to Default Project Templates.
- A truly Empty Project Template.
- Based on jQuery Mobile, new Mobile Project Template introduced.
- Support for adding controller to other project folders also.
- Task Support for Asynchronous Controllers.
- Controlling Bundling and Minification through web.config.
- Support for OAuth and OpenID logins using *DotNetOpenAuth* library.
- Support for *Windows Azure SDK* 1.6 and new releases.

ASP.NET MVC 5

Creating your first ASP.NET MVC 5 Application in 4 simple steps

- Bootstrap replaced the default MVC template.
- ASP.NET Identity for authentication and identity management.
- Authentication Filters for authenticating user by custom or third-party authentication provider.
- With the help of Filter overrides, we can now override filters on a method or controller.
- Attribute Routing is now integrated into *MVC 5*.

Life Cycle of ASP.NET MVC



ASP.NET MVC Request Life Cycle

Entity Framework Versions

ADO.NET Entity Framework Version	Features Detail
6.0	1. Async Query and Save 2. Code-Based Configuration 3. Dependency Resolution 4. Interception/SQL logging 5. Improved Connection Management 6. Improved Transaction Support

5.0	1. Enum Support in Code First and EFDesigner 2. Spatial Data Types in Code First and EF Designer 3. Table-Valued Functions 4. Multiple Diagrams per Model
4	1. Code First Migrations 2. Automatic Migrations 3. Code First development 4. Introduced DbContext API 5. Data Annotations and Fluent API Validation 6. Model-first development 7. POCO support 8. Lazy Loading 9. T4 Code Generation