**Known Types in Data Contract**

**Rameshkartik.RS**

**Table of Contents**

[KnownTypes in DataContract 3](#_Toc387320678)

[Attachment 6](#_Toc387320679)

[Summary 6](#_Toc387320680)

**WHAT’s IN THIS ARTicle?**

* Known Types in DataContract
  + Global Level
  + Service Level
  + Operation Level
* Attachment
* Summary

# KnownTypes in DataContract

WCF Data Contract describes the structure or type of data being exchanged between the client and server. In some scenarios the type of data is not known between the client and server. For Example the data what the service is sending back to the client is not the actual data contract but a derived type of it. Then How can we expect the DataContractserializer will comes to know about this relationship? Until we say this relationship to the DataContractSerializer, this deserialization issue will not be resolved. Is my explanation still not good enough in explaining the problem? Let me explain you with the source code.

public interface IOnlineShoppingService

{

[OperationContract(Name = "StockAvailability")]

bool IsStockAvailable(string sModelNumber);

[OperationContract(Name = "CalculatePrice", ProtectionLevel = System.Net.Security.ProtectionLevel.None)]

double CalculatePrice(string sModelNumber, int iQuantity, string sDeliveryLocation,int iDiscountCouponNum);

[OperationContract(Name = "GetWholeSaleDealerInfo")]

WholeSaleDealersInfo GetwholeSaleDealerInfo(string sFilterByCity);

}

[DataContract]

public class WholeSaleDealersInfoDetailed:WholeSaleDealersInfo

{

//Member declaration

}

public WholeSaleDealersInfo GetwholeSaleDealerInfo(string sFilterByCity)

{

switch (sFilterByCity)

{

case "Chennai":

if (sFilterByCity.Equals("Chennai"))

{

WholeSaleDealersInfoDetailed wsDealerDetailed = new WholeSaleDealersInfoDetailed();

wsDealerDetailed.DealerName = "Rameshkartik";

wsDealerDetailed.DealerLocation = "Chennai";

wsDealerDetailed.DealerSince = DateTime.Now;

wsDealerDetailed.DealerEmail = "rameshkartik@gmail.com";

wsDealerDetailed.DealerAddress = "North Street,Chennai";

return wsDealerDetailed;

}

break;

return null;

}

return null;

}

In the above service interface you can find a method called GetWholeSaleDealerInfo which returns the type WholeSaleDealersInfo, but the DataContractSerializer will expect the only type WholeSaleDealersInfo not the WholeSaleDealersInfoDetailed. This is what the problem, during the deserialization the actual contract will not be coming but it is a derived one. So what is the solution for this? As discussed earlier Known types in Data Contract is the right way to solve this. You can use Known Types at the different levels based on your needs

* Global Level
* Service Contract Level
* Operation Contract Level

Global Level

[KnownType(typeof(WholeSaleDealersInfoDetailed))]

[DataContract]

public class WholeSaleDealersInfo

{

}

In the above example you can see that KnownType attribute is used at the Data Contract level. It means that the type is globally known to all the service contracts,operation contracts wherever the WholeSaleDealersInfo type is used.

Service Contract Level

[ServiceKnownType(typeof(WholeSaleDealersInfoDetailed))]

public interface IOnlineShoppingService

{

[OperationContract(Name = "GetWholeSaleDealerInfo")]

WholeSaleDealersInfo GetwholeSaleDealerInfo(string sFilterByCity);

}

If you want to use this derived type as a known type for a particular service, please use the above syntax Service Known Type.

Operation Contract Level

public interface IOnlineShoppingService

{

[ServiceKnownType(typeof(WholeSaleDealersInfoDetailed))]

[OperationContract(Name = "GetWholeSaleDealerInfo")]

WholeSaleDealersInfo GetwholeSaleDealerInfo(string sFilterByCity);

}

If you want use the derived type as a known for a particular method, Please place the serviceknown type attribute on the service method as like above.

We can achieve the same in the web.config as well

# Attachment

Refer the code attachment for further details

# 

# Summary