Proxies, Channels, and Clients

Miguel A. Castro
@miguelcastro67

pluralsight
hardcore developer training

Consuming WCF Services

- **Complex process to establish communication with a hosted service**
  - Pipe establishment
  - Handshaking
  - Security exchange
  - Message transmission
- **Lots of low-level stuff (unmanaged code)**
- **This is where SOAP tooling comes in**
- **Client needs to know what is available and the data shape**
  - Data & Service Contracts
- **What you want**
  - .NET object to make it easy for clients to make a call
  - Easy operation methods
  - Abstraction of all low level complexity

Enter the Proxy

- **WCF provides the ClientBase<T> class**
  - T is the service contract
- **Proxy class inherits it**
- **Proxy also implements the service contract interface**
  - That's where the easy operation methods come from
- **Contracts assembly can be shared by client side**
  - May not always be the case
  - Dependent on application architecture
- **Proxy methods use "Channel" property from ClientBase<T>**
- **Proxy also requires endpoint information**
  - Can be fed through code
  - Usually fed through configuration

Demo Time

Contract Equivalence

- **Service and Data contracts can be housed in a shared DLL**
- **Can also reside in different binaries for service or client**
- **Which solution used depends on architectural and design considerations**
- **When using two contracts, both sides need to be equivalent**
  - Contract names and member/operation names must be the same
  - If not, can use “Name” property of attributes to change
- **Contracts namespace must match**
- **Use “Namespace” property to set**
  - Can also use **ContractNamespace** assembly attribute

Demo Time

Channel Factory

- **Internally, proxies wrap the creating of a "channel"**
  - Provide developer-friendly class
- **Remember a proxy is an implementation of the service contract**
- **A service contract implementation can be obtained without proxy**
  - Kind of a virtual proxy
  - No actual proxy class
- **Channel factory creates a proxy for you**
  - Tied to the service contract
  - Can use config or go config-less

Demo Time

Version Tolerance

- **DataContractSerializer provides version tolerance**
- **Two equivalent contracts may be "off" in their properties**
- **Use IExtensibleDataObject interface**
  - ExtensionData property (ExtensionDataObject type)
- **All you need to do is implement it in data contracts**
  - Property can be use short-form syntax
- **WCF will use property to store extra property values for potential pass-through scenarios**