**Hosting**

호스팅은 WCF 서비스를 클라이언트에서 사용할 수 있게끔 해주는 중요한 작업이죠

WCF는 여러 가지의 방법으로 호스팅을 할 수 있는 장점이 있습니다. 이는 제가 처음 WCF를 소개할 때도 언급했던 내용이었구요.

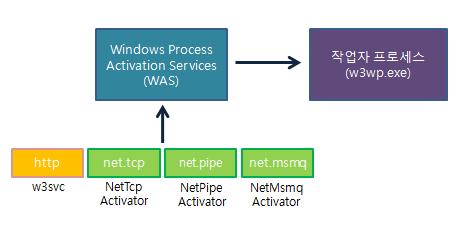
그래서, 지금부터는 WCF 서비스를 호스팅할 수 있는 방법과 호스팅할 때 사용할 수 있는 기능에 대해서 알아볼까 합니다.  
  
지금까지 8번의 포스팅을 진행하면서 예제로 사용했던 WCF 서비스들은 모두 **Self Hosting**이었습니다.

|  |
| --- |
| [Selft Hosting]  특별한 무언가가 있는 것은 아니구요, **System.ServiceModel 네임스페이스에 정의 되어있는 ServiceHost 클래스를 사용하여 직접 서비스를 호스팅한 것**을 의미합니다.  아시겠죠? 지금까지 제가 사용했던 예제들은 모두 콘솔 어플리케이션 내에서 직접 ServiceHost 클래스를 이용하여 서비스를 호스팅했으니 모두 Self Hosting이 되는 것입니다. 물론, 윈도우 폼 프로그램, WPF 프로그램 그리고, 윈도우 서비스를 이용하여 WCF 서비스를 호스팅할 수 있는데, 이것들 역시 셀프 호스팅(Self Hosting) 인 것입니다. |

Self Hosting 외에도 WCF 서비스를 한번이라도 개발하신 분이라면 아마 사용했을 법한 **IIS 호스팅**이 있습니다. 그리고 IIS7에서 사용할 수 있는 **WAS 호스팅**이 있습니다.  
  
IIS 호스팅은 웹 서비스 처럼 IIS를 이용한 호스팅이라 특별한 것도 없고, WCF 서비스를 호스팅하기 위한 가장 쉬운 방법이기도 하구요, 그래서 따로 포스팅을 하지 않아도 될 것 같습니다.  
  
그래서, 이번 포스팅은 WAS 호스팅에 대해서 좀 더 자세히 알아보려 합니다.

**WAS(Windows Process Activation Service) 를 이용한 호스팅**  
WAS는 Windows Process Activation Service의 약자로 IIS 7.0의 기본 구성 요소로서 **HTTP 이외의 프로토콜(TCP, MSMQ, Named Pipes)을 사용한 서비스를 호스팅할 수 있게 해주는 역할을 수행**합니다.

그럼, 간단하게 WAS가 어떻게 HTTP 이외의 프로토콜을 사용할 수 있게 하는지 알아볼까요?  
다음은 WAS 의 아키텍처를 간략하게 표현한 그림입니다.

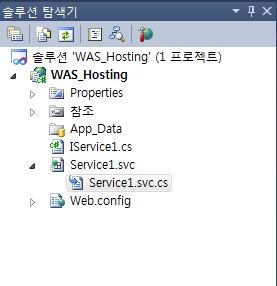
[](http://cfile30.uf.tistory.com/original/207098244C02328A6DAB28)

서버는 서버로 어떤 요청이 들어왔을 때, 그 요청을 처리할 수 있는 수신기를 가지고 있으며, 이 수신기(Listener)는 각 요청의 프로토콜에 맞는 수신기 어댑터(TCP 수신기 어댑터, MSMQ 수신기 어댑터, Named Pipe 수신기 어댑터)로 요청을 보냅니다.

수신기와 WAS 사이에는 수신기 어댑터 인터페이스가 존재하는데, 이를 이용해 각 프로토콜의 수신기 어댑터는 전달받은 요청을 WAS로 보낼 수 있게 되는 거죠. 그리고 WAS 에서는 각 요청에 맞는 응용 프로그램 인스턴스를 생성하기 위해 작업자 프로세스로 요청을 전달하게 되는 것입니다.  
  
이러한 일련의 작업들을 통해 WAS는 HTTP 이외의 프로토콜을 이용한 서비스를 가능하게 하는 것입니다.  
  
그럼, WAS를 이용하여 호스팅을 해보도록 하겠습니다.  
  
WAS는 Vista 이상의 Windows 에서는 IIS를 설치할 때 기본으로 함께 설치가 됩니다. 이 때는 HTTP 수신기 어댑터가 설치 되구요, .NET 3.5 이상의 버전이 설치될 때 TCP, MSMQ, named pipe 수신기 어댑터가 설치 됩니다.  
  
WAS를 이용한 서비스를 만들기 위해 가장 먼저 해야할 것은 IIS를 이용한 호스팅을 하는 WCF 서비스 솔루션을 만드는 것입니다. WAS를 이용한 서비스를 만드는 특별한 방법이 있는 것은 아니며, 기본적으로 WAS는 IIS 7.0의 요소이기 때문에 우선 IIS를 이용한 서비스를 만드는 것이 가장 먼저 해야할 일인 것입니다.

Visual Studio 2010에서 **"WCF 서비스 응용 프로그램"** 솔루션을 선택합니다.

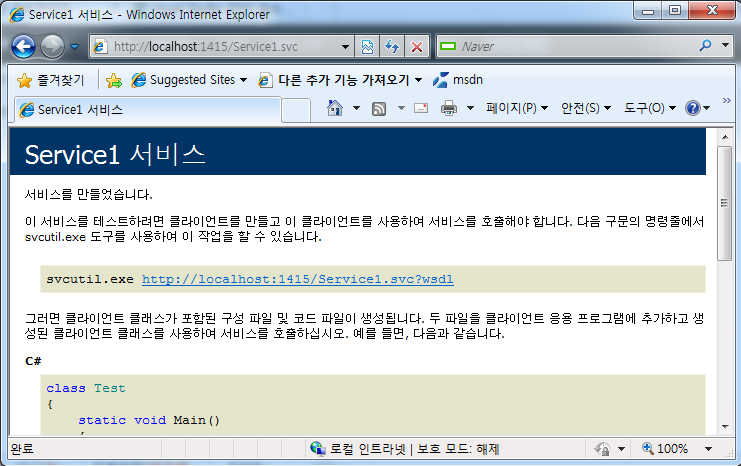
이렇게 새로운 솔루션을 생성하면, IIS를 이용한 WCF 서비스를 만들 수 있습니다. 기본적으로 Service1.svc 라는 파일이 만들어지는데 이 파일이 WCF 서비스의 로직이 담기는 핵심 파일입니다.(솔루션의 구조는 다음 그림과 같은 모습을 하고 있습니다.)

[](http://cfile3.uf.tistory.com/original/1459B6254C03B1926B9053)

여기서 서비스의 기능을 바꾸려면 어떤 파일을 수정해야 할까요? 바로 찾을 수 있으시겠죠? IService1.cs 와 Service1.svc.cs 파일을 바꾸어야 합니다.

물론, 이번 포스팅에서의 주제는 WAS를 이용한 호스팅이기 때문에 자동으로 만들어지는 서비스의 로직을 굳이 바꿀 필요는 없을 것 같습니다.

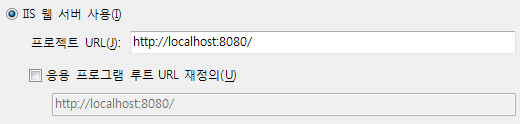
그래~서~!! 저는 따로 서비스의 기능을 바꾸지 않고 자동으로 만들어진 서비스를 그대로 이용해 보겠습니다.   
솔루션을 만든 후 바로 Ctrl+F5 키를 눌러서 실행을 시켜보도록 하죠~   
  
ASP.NET 솔루션을 실행시킨 것 처럼 ASP.NET Development Server가 시작되고, 웹 브라우저를 이용해 svc 파일을 탐색해 보면 다음과 같이 WCF 서비스 안내 페이지를 만날 수 있을 것입니다.

[](http://cfile1.uf.tistory.com/original/136A0E0C4C03B539372872)

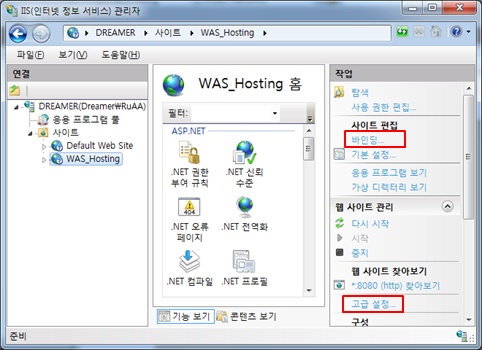
IIS를 이용한 호스팅은,. 정말 너무나도 쉽습니다. ServiceHost 클래스를 사용하지도 않고, 자동으로 만들어진 파일 그대로 실행을 해도 무리없이 실행이 되는 것을 알 수 있습니다.   
  
오늘 포스팅의 목적인 WAS를 이용한 호스팅을 구현하기 위해 몇가지 설정을 해보도록 하겠습니다.

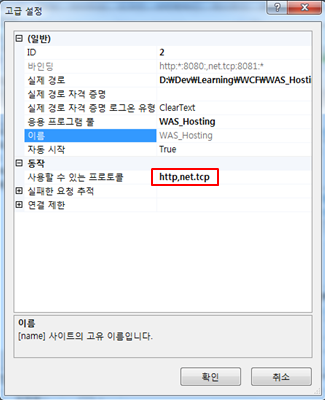
바로 이전에 실행했던 서비스는 당연한 이야기겠지만, IIS를 이용한 호스팅이었고, 이는 HTTP 프로토콜을 사용하고 있습니다.

WAS 호스팅을 실행하기 위해서 우선 서비스 솔루션을 ASP.NET Development Server 가 아닌 IIS에서 실행이 되도록 바꾸어야 합니다. 그래서 솔루션 속성창 "웹" 메뉴에서 **"IIS 웹 서버 사용"**으로 설정해줍니다. (아래 그림 참조)

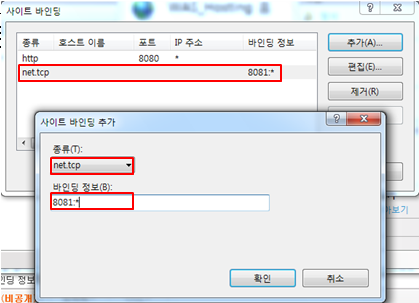
[](http://cfile10.uf.tistory.com/original/1743F2024C0636D944B2F4)

IIS 관리자를 실행시키고, 이 사이트가 돌아갈 수 있도록 새로운 사이트를 만들어 주는 것도 잊으시면 안됩니다~ 이 내용은 다들 알고 계시리라 생각하고 생략하겠습니다 ^^  
  
WAS를 이용한 호스팅의 목적은 HTTP 프로토콜 이외의 다른 프로토콜을 사용하기 위함입니다. 그래서 저는 이번 예제에서 TCP 프로토콜을 사용할 수 있는 서비스를 만들어보려 합니다. TCP 프로토콜을 사용할 수 있도록 하기 위해선 IIS 관리자에서 설정을 변경해주어야 할 것이 몇 개 있습니다.  
  
IIS 관리자에서 해당하는 사이트를 선택하고 오른쪽에 위치한 **"고급설정"**을 클릭하여 설정 창을 띄웁니다. 고급설정 창에서 "사용할 수 있는 프로토콜"에 아래 그림과 같이 **"net.tcp"**를 추가하여 줍니다.

[](http://cfile4.uf.tistory.com/original/143943154C06437B7F5DC7)

[](http://cfile22.uf.tistory.com/original/123E4F214C063D0F8B36F8)

그리고, 한 가지 더 설정해주어야 할 것이 있는데, 해당 사이트와 특정 프로토콜에 대한 바인딩입니다. 사이트 작업 메뉴에서 "바인딩"을 클릭하여 사이트 바인딩 창을 띄웁니다. 그리고, 추가 버튼을 클릭하여 다음 그림과 같이 net.tcp 프로토콜에 대한 바인딩 정보를 입력하여 줍니다.

[](http://cfile6.uf.tistory.com/original/12341C014C06445E026A3E)

여기까지 설정이 완료되면, WAS를 이용하여 호스팅을 하기 위한 IIS 사이트 설정이 모두 끝이 납니다.

이제, 마지막으로 앞에서 만든 WCF 서비스에 TCP 프로토콜을 사용할 수 있도록 netTcpBinding을 사용하는 엔드포인트를 추가시켜주어야 합니다. WCF 서비스 솔루션에 있는 web.config 파일을 열어보죠.  
  
Visual Studio 2010을 이용해 서비스를 만드셨다면, web.config의 내용이 조금 달라졌다고 느끼실 것 같습니다. 사실, 내용이 많이 달라진 것은 아니고, 그 전 버전에 비해 web.config의 내용이 아주 많이 간략화되었습니다. 매우 매우 심플해졌죠. 이 내용에 대해서 나중에 따로 포스팅을 할 기회가 있을 것이라 생각됩니다.  
  
엔드 포인트를 다음과 같이 포함시켜 보겠습니다.

<system.serviceModel>

<services>

<service name="WAS\_Hosting.Service1">

<endpoint address="mex" binding="mexTcpBinding" contract="IMetadataExchange" />

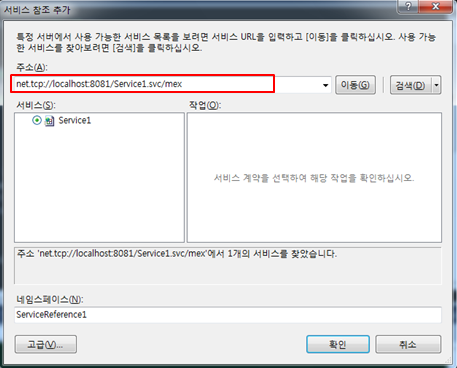
<endpoint binding="netTcpBinding" contract="WAS\_Hosting.IService1" />

</service>

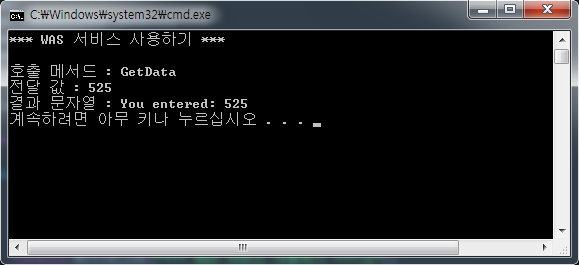
</services>  
    ...  
</system.serviceModel>

**service 태그의 name 속성과 endpoint 태그의 contract 속성은 WCF 서비스의 네임스페이스와 클래스 이름을 잘 확인하여 설정해주셔야 합니다.**  
  
IMetadataExchange를 컨트랙트로 사용하는 엔드포인트도 같이 설정을 해주는 걸 볼 수 있습니다. 이는 서비스의 메타 데이터에 접근할 수 있는 엔드 포인트란 것을 알 수 있으실 겁니다.

이 서비스가 제대로 서비스가 되는지 확인해보기 위해 콘솔 프로젝트를 새로 만들고 서비스 참조를 통해 우리가 만들었던 WCF 서비스를 참조해보도록 하겠습니다.  
  
IIS 관리자에서 저는 net.tcp에 대한 바이딩을 할 때 **포트번호를 8081**로 설정했었습니다. 따라서 그림에서 볼 수 있듯이 "net.tcp://localhost:8081/Service1.svc" 주소로 WCF 서비스에 접근할 수 있습니다.

[](http://cfile1.uf.tistory.com/original/157FD60F4C064C843F7B18)

서비스를 참조한 후에 서비스를 사용하는 방법은 특별한 것이 없습니다. 예전에 셀프 호스팅 서비스를 사용했던 것과 같은 방법으로 사용하면 되기에 클라이언트 쪽 코드는 생략하고, 결과 화면만 첨부하겠습니다.

[](http://cfile22.uf.tistory.com/original/1471F30C4C064E9D91688C)

WCF 서비스가 아주 잘 동작하는 것을 확인할 수 있습니다. 유후~ ^^