**용어 정리**[**(http://msdn.microsoft.com/ko-kr/library/dd456781.aspx)**](http://msdn.microsoft.com/ko-kr/library/dd456781.aspx)

|  |  |
| --- | --- |
| 용어 | 정의 |
| 주소(address) | 메시지가 수신될 위치를 지정하며, URI(Uniform Resource Identifier)로 지정됩니다. URI 스키마 부분은 HTTP 및 TCP와 같은 주소에 도달하는 데 사용할 전송 메커니즘 이름을 지정합니다. URI의 계층적 부분에는 전송 메커니즘에 따라 다른 고유한 위치 형식이 포함됩니다. |
| 응용 프로그램 끝점(application endpoint) | 응용 프로그램에 의해 노출된 끝점이며 응용 프로그램이 구현하는 서비스 계약에 해당합니다. |
| 동작(behavior) | 동작은 서비스, 끝점, 특정 작업 또는 클라이언트의 다양한 런타임 측면을 제어하는 구성 요소입니다. 동작은 범위에 따라 그룹화됩니다. 일반 동작은 모든 끝점에 전역적으로 영향을 주고 서비스 동작은 서비스 관련 측면에만 영향을 주며, 끝점 동작은 끝점 관련 속성에만 영향을 주고 작업 수준 동작은 특정 작업에 영향을 줍니다. |
| 바인딩(binding) | WCF 서비스와 통신하는 데 사용되는 통신 프로토콜을 정의하며, 통신 인프라를 만들기 위해 다른 요소의 맨 위에서 요소를 스택하는 바인딩 요소라는 구성 요소 집합으로 생성됩니다. 끝점을 참조하십시오. |
| 채널(channel) | 바인딩 요소의 구체적인 구현입니다. 바인딩은 구성을 나타내고 채널은 해당 구성과 관련된 구현입니다. 따라서 각 바인딩 요소마다 관련된 채널이 있습니다. 채널은 다른 채널 맨 위에 스택되어 바인딩의 구체적인 구현인 채널 스택을 만듭니다. |
| 클레임 기반 보안(claims-based security) | 클레임을 기반으로 보호된 리소스에 대한 허가된 액세스를 제공합니다. |
| 클라이언트 응용 프로그램(client application) | 메시지를 하나 이상의 끝점과 교환하는 프로그램입니다. 클라이언트 응용 프로그램은 WCF 클라이언트의 인스턴스를 만들고 WCF 클라이언트의 메서드를 호출하여 시작합니다. 하나의 응용 프로그램이 클라이언트 및 서비스 역할을 모두 수행할 수 있다는 점에 주의하십시오. |
| 코딩(coding) | 개발자가 서비스 또는 클라이언트의 모든 구성 요소에 대해 엄격한 제어를 유지할 수 있도록 하며, 구성을 통해 수행된 모든 설정을 검사하고 필요한 경우 코드를 통해 재지정할 수 있습니다. 코딩이나 구성을 사용하여, 또는 두 가지를 모두 사용하여 응용 프로그램을 제어할 수 있습니다. |
| 구성(configuration) | 구성은 개발자 이외의 다른 담당자(예: 네트워크 관리자)가 코드를 기록한 후 다시 컴파일할 필요 없이 클라이언트 및 서비스 매개 변수를 설정할 수 있는 이점이 있습니다. 구성을 사용하면 끝점 주소와 같은 값을 설정할 수 있을 뿐만 아니라 끝점, 바인딩 및 동작을 추가하여 보다 세부적인 제어를 수행할 수 있습니다. 코딩이나 구성을 사용하여, 또는 두 가지를 모두 사용하여 응용 프로그램을 제어할 수 있습니다. |
| 계약(contract) | 계약은 계약의 특정 형식에 대한 지원 사양입니다. 예를 들어, 서비스 계약은 작업 그룹의 사양입니다. WCF에서 계약은 System.ServiceModel.Description 네임스페이스에 있는 설명 개체에 미러링되는 계층 구조를 가집니다. 서비스 계약은 WCF에서 가장 큰 계약 범위입니다. 서비스 계약에 있는 각 서비스 작업에는 작업에서 교환할 수 있는 메시지(오류 메시지 포함) 및 교환 방향을 지정하는 작업 계약이 포함됩니다. 작업의 각 메시지에는 메시지 계약, SOAP 메시지 봉투의 구조에 대한 사양이 포함되고, 각 메시지 계약에는 메시지에 포함된 데이터 구조를 지정하는 데이터 계약이 포함됩니다. |
| 데이터 계약(data contract) | 서비스가 사용하는 데이터 형식은 다른 서비스와 해당 서비스가 상호 운용될 수 있도록 메타데이터로 설명해야 합니다. 데이터 형식에 대한 설명을 데이터 계약이라고 하며 형식은 예를 들어 매개 변수 또는 반환 형식과 같이 메시지 일부로 사용할 수 있습니다. 서비스가 간단한 형식만 사용하는 경우 명시적으로 데이터 계약을 사용할 필요가 없습니다. |
| 선언적 응용 프로그램(declarative application) | 명령적 지시를 실행하지 않고도 런타임에 작성할 수 있도록 충분히 설명된 응용 프로그램입니다. |
| 끝점(endpoint) | WCF 서비스와 통신하는 데 사용되는 주소, 바인딩 및 계약으로 구성됩니다. |
| 끝점 주소(endpoint address) | 서비스의 각 끝점마다 고유한 끝점 주소를 만들거나 특정한 조건으로 여러 끝점 간에 주소를 공유할 수 있도록 합니다. |
| 오류 계약(fault contract) | 오류 계약은 서비스 작업과 연결하여 호출자에게 반환될 수 있는 오류를 나타낼 수 있습니다. 작업에는 작업과 관련된 오류가 0개 이상 포함될 수 있습니다. 이러한 오류는 프로그래밍 모델에서 예외로 모델링된 SOAP 오류입니다. 예외는 클라이언트에 보낼 수 있는 SOAP 오류로 변환됩니다. |
| 호스팅(hosting) | 일부 프로세스의 경우 서비스를 호스팅해야 합니다. 호스트는 서비스 수명을 제어하는 응용 프로그램입니다. 서비스는 기존 호스팅 프로세스에 의해 자체 호스팅되거나 관리될 수 있습니다. |
| 호스팅 프로세스(hosting process) | 호스팅 프로세스는 서비스를 호스팅하기 위해 디자인된 응용 프로그램입니다. 여기에는 IIS(인터넷 정보 서비스), WAS(Windows Activation Services) 및 Windows Services가 있습니다. 이러한 호스팅된 시나리오에서 호스트는 서비스 수명을 제어합니다. 예를 들어, IIS를 사용하는 경우 서비스 어셈블리 및 구성 파일이 포함된 가상 디렉터리를 설정할 수 있습니다. 메시지를 수신하면 IIS는 서비스를 시작하고 수명을 제어합니다. |
| 시작 작업(initiating operation) | 새 세션의 첫 번째 작업으로 호출되는 작업입니다. 하나 이상의 작업을 호출한 후에만 시작 작업이 아닌 작업을 호출할 수 있습니다. |
| 인스턴스 만들기 모델(instancing model) | 서비스에는 인스턴스 만들기 모델이 포함되어 있습니다. 인스턴스 만들기 모델에는 세 가지가 있습니다. 즉, 단일 CLR 개체가 모든 클라이언트에 서비스를 제공하는 "단일" 모델, 각 클라이언트 호출을 처리하기 위해 새로운 CLR 개체를 만드는 "호출별" 모델 및 각 개별 세션에 대해 CLR 개체 집합을 만드는 "세션별" 모델입니다. 인스턴스 만들기 모델은 응용 프로그램 요구 사항 및 예상되는 서비스 사용 패턴에 따라 선택합니다. |
| 메시지(message) | 메시지는 본문 및 헤더를 포함하여 여러 부분으로 구성될 수 있는 독립적인 데이터 단위입니다. |
| 메시지 계약(message contract) | 메시지 계약은 메시지 형식을 설명합니다. 예를 들어, 메시지 요소가 헤더 또는 해당 본문에 삽입되어야 할지 여부, 메시지 요소에 적용되어야 하는 보안 수준 등을 선언합니다. |
| 메시지 보안 모드(message security mode) | 메시지 보안 모드는 하나 이상의 보안 사양을 구현하여 보안을 제공하도록 지정합니다. 각 메시지에는 전송 중에 보안을 제공하고 수신자가 변조를 감지하거나 메시지를 해독하는 데 필요한 메커니즘이 포함되어 있습니다. 따라서 보안은 모든 메시지 내에 캡슐화되어 여러 홉 전반에 걸쳐 종단 간 보안을 제공합니다. 보안 정보가 메시지의 일부이기 때문에 여러 종류의 자격 증명을 메시지에 포함시킬 수도 있습니다. 이러한 자격 증명을 클레임이라고 합니다. 또한 이러한 접근 방식은 원본 및 대상 간 여러 전송을 포함하여 메시지가 전송을 통해 안전하게 전달될 수 있는 이점이 있습니다. 그러나 관련 암호화 메커니즘이 복잡하기 때문에 성능이 떨어질 수 있다는 단점이 있습니다. |
| 메타데이터(metadata) | 서비스 메타데이터는 서비스와 통신하기 위해 외부 엔터티가 이해해야 하는 서비스 특성을 설명합니다. ServiceModel Metadata 유틸리티 도구(Svcutil.exe)에서 메타데이터를 사용하여 서비스와 상호 작용하는 데 클라이언트 응용 프로그램이 사용할 수 있는 WCF 클라이언트 및 관련 구성을 생성할 수 있습니다. 서비스에 의해 노출된 메타데이터에는 서비스의 데이터 계약을 정의한 XML 스키마 문서와 서비스의 메서드를 설명하는 WSDL 문서가 포함되어 있습니다. 사용할 경우 서비스와 끝점을 검사하여 WCF에 의해 서비스 메타데이터가 자동으로 생성됩니다. 서비스에서 메타데이터를 게시하려면 명시적으로 메타데이터 동작을 사용해야 합니다. |
| 작업 계약(operation contract) | 작업 계약은 매개 변수를 정의하고 작업 형식을 반환합니다. 서비스 계약을 정의하는 인터페이스를 만드는 경우 T:System.ServiceModel.OperationContractAttribute 특성을 계약의 일부인 각 메서드 정의에 적용하여 작업 계약을 나타냅니다. 작업은 단일 메시지를 가져오고 단일 메시지를 반환하거나 형식 집합을 가져오고 형식을 반환하는 방법으로 모델링할 수 있습니다. 후자의 경우 시스템에서 해당 작업에 대해 교환되는 메시지 형식을 결정합니다. |
| 프로젝션 | 연결되어 있는 데이터의 표현입니다. 예를 들어 SOAP 프로젝션은 메시지를 SOAP 봉투로 보내고 웹 프로젝션은 메시지를 JSON 형식으로 보냅니다. |
| 보안 | WCF의 보안에는 기밀성(도청을 방지하기 위해 메시지 암호화), 무결성(메시지 변조를 감지하기 위한 방법), 인증(서버 및 클라이언트의 유효성 검사를 위한 방법) 및 권한 부여(리소스 액세스 권한 제어)가 포함됩니다. 이러한 기능은 HTTP를 통한 TLS(HTTPS라고도 함)와 같이 기존 보안 메커니즘을 활용하거나 다양한 WS-\* 보안 사양 중 하나 이상을 구현하여 제공합니다. |
| 자체 호스팅 서비스(self-hosted service) | 자체 호스팅 서비스는 개발자가 만든 프로세스 응용 프로그램 내에서 실행되는 서비스입니다. 개발자는 수명을 제어하고 서비스 속성을 설정하며 서비스를 열고(수신 대기 모드로 설정된 경우) 서비스를 닫습니다. |
| 서비스(service) | 하나 이상의 끝점을 노출하는 프로그램 또는 프로세스이며 각 끝점은 하나 이상의 작업을 노출합니다. |
| 서비스 계약(service contract) | 서비스 계약은 서로 연관된 여러 작업을 하나의 기능 단위로 묶습니다. 계약은 서비스의 네임스페이스, 해당 콜백 계약 및 기타 설정과 같은 서비스 수준 설정을 정의할 수 있습니다. 대부분의 경우 계약은 선택한 프로그래밍 언어로 인터페이스를 만들고 T:System.ServiceModel.ServiceContractAttribute 특성을 인터페이스에 적용하여 정의합니다. 실제 서비스 코드는 인터페이스를 구현한 결과입니다. |
| 서비스 작업(service operation) | 서비스 작업은 작업의 기능을 구현하는 서비스 코드에서 정의된 프로시저입니다. 이 작업은 WCF 클라이언트에서 메서드로 클라이언트에 노출됩니다. 메서드는 값을 반환하고 선택적 인수 개수를 가져오거나, 인수를 가져오지 않고 응답을 반환하지 않을 수 있습니다. 예를 들어, "Hello"로 기능하는 작업을 클라이언트의 존재에 대한 알림으로 사용하고 일련의 작업을 시작할 수 있습니다. |
| 시스템 제공 바인딩(system-provided binding) | WCF에는 다양한 시스템 제공 바인딩이 포함되어 있습니다. 이러한 바인딩은 특정 시나리오에 맞게 최적화된 바인딩 요소의 컬렉션입니다. 예를 들어, T:System.ServiceModel.WSHttpBinding은 다양한 WS-\* 사양을 구현하는 서비스와의 상호 운용성을 위해 디자인되었습니다. 이러한 바인딩은 특정 시나리오에 제대로 적용될 수 있는 옵션만 제공하기 때문에 시간이 절약됩니다. 이러한 바인딩 중 하나가 요구 사항을 충족하지 않을 경우 사용자 지정 바인딩을 만들 수 있습니다. |
| 종료 작업(terminating operation) | 기존 세션에서 마지막 메시지로 호출되는 작업입니다. 기본적으로 WCF에서는 서비스와 연결된 세션이 닫힌 후에 서비스 개체 및 컨텍스트를 재활용합니다. |
| 전송 보안 모드(transport security mode) | 보안은 세 가지 모드 즉, 전송 모드, 메시지 보안 모드 및 메시지 자격 증명을 사용한 전송 모드 중 하나를 통해 제공할 수 있습니다. 전송 보안 모드는 전송 계층 메커니즘(예: HTTPS)에 의해 제공되는 기밀성, 무결성 및 인증을 지정합니다. HTTPS와 같은 전송을 사용하면 이 모드는 성능이 효율적이며 인터넷에서 널리 사용되기 때문에 이해하기 쉽습니다. 단점은 이러한 종류의 보안이 통신 경로의 각 홉에서 개별적으로 적용되기 때문에 통신이 "메시지 가로채기(man in the middle)" 공격을 받기 쉽습니다. |
| 메시지 자격 증명을 사용한 전송 보안 모드(transport with message credential security mode) | 이 모드는 전송 계층을 사용하여 기밀성, 인증 및 메시지 무결성을 제공하는 동시에 각 메시지에는 메시지 수신자에게 필요한 여러 자격 증명(클레임)이 포함될 수 있습니다. |
| 형식 변환기(type converter) | CLR 형식은 하나 이상의 System.ComponentModel.TypeConverter 파생 형식과 연결할 수 있습니다. 이 파생 형식을 사용하면 CLR 형식 인스턴스와 다른 형식 인스턴스 간에 변환할 수 있습니다. 형식 변환기는 System.ComponentModel.TypeConverterAttribute 특성을 사용하여 CLR 형식과 연결됩니다. TypeConverterAttribute는 CLR 형식 또는 속성에서 직접 지정할 수 있습니다. 속성에 대해 지정된 형식 변환기는 항상 속성의 CLR 형식에 대해 지정된 형식 변환기보다 우선적으로 적용됩니다. |
| WCF 클라이언트(WCF client) | 서비스 작업을 메서드로(Visual Basic 또는 Visual C#과 같이 선택한 .NET Framework 프로그래밍 언어로 표시) 노출하는 클라이언트 응용 프로그램 구문입니다. 모든 응용 프로그램은 서비스를 호스팅하는 응용 프로그램을 포함하여 WCF 클라이언트를 호스팅할 수 있습니다. 따라서 다른 서비스의 WCF 클라이언트를 포함하는 서비스를 만들 수 있습니다. WCF 클라이언트는 ServiceModel Metadata 유틸리티 도구(Svcutil.exe)를 사용하고 메타데이터를 게시하는 실행 중인 서비스에서 선택하여 자동으로 만들 수 있습니다. |
| 워크플로 서비스 | 워크플로 서비스는 워크플로로 구현되는 WCF 서비스입니다. 워크플로에는 WCF 메시지를 보내거나 받는 메시징 활동이 포함됩니다. |
| WS-\* | WS-Security, WS-ReliableMessaging 등과 같이 WCF에서 구현되고 계속 업데이트되는 WS(웹 서비스) 사양 집합의 약어입니다. |
| XAML | eXtensible Application Markup Language의 약어입니다. |
| XAML 스키마 | XAML에서 사용자 지정 형식을 정의하는 데 사용되는 마크업 스키마입니다. |