ASP.NET Web Services

José Antônio da Cunha

IFRN

ASP.NET Web Services

Algumas das aplicações mais interessantes Silverlight tem backbone de código escondido do lado do servidor. Eles podem chamar um servidor Web para recuperar dados de um banco de dados, realizar a autenticação, armazenamento de dados em um repositório central, apresentar uma tarefa demorada, ou executar qualquer número de outras tarefas que não são possíveis com o código do lado do cliente sozinho.

ASP.NET Web Services

Construíndo Web Services para o Silverlight

Sem dúvida, a maneira mais eficaz para uma aplicação Silverlight rodar código do servidor é através de web services. A idéia básica é simples: você incluir um Web Service em seu website com ASP.NET, e seu aplicativo Silverlight chama os métodos desse serviço.

Aplicações Silverlight podem chamar serviços web tradicionais ASP.NET (. serviços asmx), bem como serviços WCF, que são o padrão mais recente.

ASP.NET Web Services

Criando um Web Services

Para criar um serviço WCF no Visual Studio, clique com botão direito do seu site ASP.NET no Solution Explorer e escolha Add New Item. Escolha o **Silverlight-enabled WCF** Service template. Entre com o nome e clique em Add.

ASP.NET Web Services

Quando você adiciona um novo WCF, Visual Studio cria dois arquivos:

•O serviço Endpoint: O serviço Endpoint tem a extensão **.svc** e é colocado na pasta raiz do site. Este arquivo não contém nenhum código – ele inclui uma linha de marcação que diz ASP.NET onde encontrar o código correspondente serviço web.
•O códig: o arquivo que contém o código é colocado na pasta App_Code e tem a extensão .cs (se você estiver trabalhando com o C#). Um arquivo de código com uma classe que implementa a interface de serviço e fornece o código real para o seu serviço de web.

ASP.NET Web Services

O arquivo de código para o seu serviço web começa com dois atributos. O atributo **ServiceContract** indica que define um contrato de serviços - em outras palavras, um conjunto de métodos que pretende expor aos chamadores remoto como parte de um serviço. O atributo **AspNetCompatibilityRequirements** indica que ele terá acesso a recursos da plataforma ASP.NET, como o estado da sessão:

```
using System;
...
namespace WebServiceWCF.Web
{
    [ServiceContract(Namespace = "")]
    [AspNetCompatibilityRequirements(RequirementsMode =
AspNetCompatibilityRequirementsMode.Allowed)]
    public class TestService
    {
        [OperationContract]
        public void DoWork()
        {
            // Add your operation implementation here
            return;
        }

        // Add more operations here and mark them with [OperationContract]
    }
}
```

ASP.NET Web Services

Adicionando referência a um serviço (Web Service)

Você consome um serviço web em uma aplicação em Silverlight da mesma maneira que você consume em qualquer outro aplicativo. O primeiro passo é criar uma classe proxy adicionando um serviço de referência do Visual Studio.

Para adicionar uma referência ao serviço web, seguia estes passos:

1. Clique com o botão direito do mouse no projeto Silverlight no Solution Explorer, e escolha **Add Service Reference**. Veja a **Figura 1**:
2. Em Address, aponte para o Web Service, e clique em GO. Se você não sabe o endereço, clique em Discover, para o sistema descobrir pra você o Web Service.
3. Em Namespace digite o nome que você que dá ao Web service.
4. Clique OK.

ASP.NET Web Services

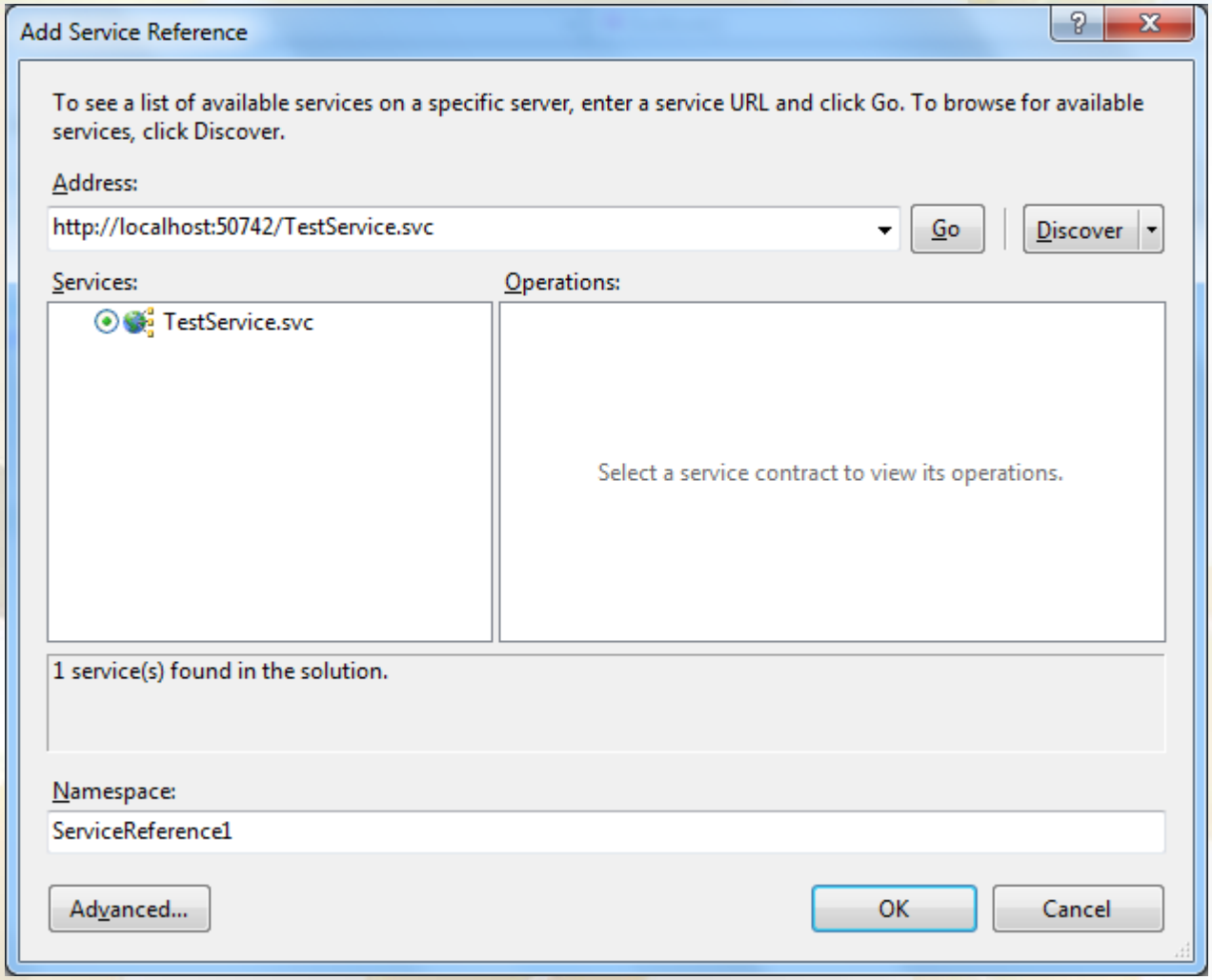

Figura 1: Adicionando referência a um serviço (Web Service)

ASP.NET Web Services

Quando você adiciona uma referência de serviço, o Visual Studio cria uma classe proxy - uma classe que você pode interagir para chamar o serviço web. A classe de proxy é nomeado após a classe original do serviço Web com a palavra **Cliente** adicionado no final. Por exemplo, ao adicionar uma referência para o **TestService**, Visual Studio cria uma classe proxy chamado **TestServiceClient**. A classe de proxy contém métodos que permitem a você disparar as chamadas de serviço adequado na web, e todos os eventos que lhe permitem receber os resultados. Ela cuida do trabalho pesado (criando a mensagem de solicitação, enviando-nos uma solicitação HTTP, obtendo a resposta e, em seguida notificando o seu código).

ASP.NET Web Services

Chamando o Web Service

Para usar a classe proxy, inicie importando o namespace que especifica a referencia do seu web service. Se você assumir que sua referencia é MyWebServer. Então você deve ter esta instrução:

```
using MySilverlightProject.MyWebServer;
```

No Silverlight, todas as chamadas de serviço Web deve ser assíncrona. Isso significa que você chama um método para iniciar a chamada (e enviar o pedido). Esse método retorna imediatamente. Seu código pode continuar a executar outras tarefas, ou o usuário pode continuar a interagir com o aplicativo. Quando a resposta é recebida, a classe de proxy dispara um evento de classe correspondente proxy, que é chamado na forma MethodNameCompleted. Você deve manipular esse evento para processar os resultados.

ASP.NET Web Services

Aqui está como chamar o método TestService.GetServerTime():

```
//Cria um proxy
TestServiceClient proxy = new TestServiceClient();

//Anexar um manipulador de eventos para o evento completed.
Proxy.GetServerTimeCompleted += New
EventHandler<GetServerTimeCompletedEventArgs>(GetServerTimeComp
leted);

//Inicia a chamada ao serviço web
Proxy.GetServerTimeAsync();
```

ASP.NET Web Services

Aqui está um manipulador de eventos que lê o resultado (a data e a hora atuais no servidor) e exibi-os em um TextBlock:

```
 private void GetServerTimeCompleted(object sender,
GetServerTimeCompletedEventArgs e)
{
    try
    {
        lblTime.Text = e.Result.ToLongTimeString();
    }
    catch (Excption err)
    {
        lblTime.Text = “Error contacting web service”;
    }
}
```