

Hosting do serviço

Para se tornar ativo, um serviço precisa ser disponibilizado. Este processo nós chamamos de **hosting**.

Para disponibilizar o nosso serviço, vamos precisar criar uma aplicação console que o inicia para nós. Para isso, vamos no nosso *Solution Explorer* e clicar com o botão direito do mouse em *Solution 'Passagens'* e adicionar um novo projeto, selecionando *Add -> New Project*. Em *Visual C# -> Windows Desktop*, vamos criar uma nova *Console Application*, chamada de **Hosting**.

Após criar o projeto precisamos adicionar nele uma referência que define o nosso serviço. Para isso, novamente no *Solution Explorer*, dentro de *Hosting*, vamos clicar com o botão direito do mouse em *References* e selecionar *Add -> Reference*. O *Reference Manager* irá abrir, e no menu da esquerda, na aba *Solution*, selecionaremos o projeto do serviço; e na aba *Assemblies* selecionaremos **System.ServiceModel**.

Agora com todas as referências devidamente importadas, podemos começar a criar o código que irá "subir" o nosso serviço, para isso vamos utilizar a classe `ServiceHost`, que está disponível no *namespace* **System.ServiceModel**. No construtor, esta classe pede uma referência ao serviço que ela irá subir:

```
using System.ServiceModel;
using Passagens;

class Program
{
    static void Main(string[] args)
    {
        ServiceHost host = new ServiceHost(typeof(ClienteService));
    }
}
```

Agora, para iniciar e finalizar o serviço, precisamos apenas chamar o método `Open` e `Close`, respectivamente. Algo como:

```
using System.ServiceModel;
using Passagens;

class Program
{
    static void Main(string[] args)
    {
        ServiceHost host = new ServiceHost(typeof(ClienteService));

        host.Open();
        Console.ReadLine();
        host.Close();
    }
}
```

Como subir um serviço é algo que pode acontecer um erro inesperado, vamos colocar esse bloco de código em um `try catch` e caso alguma exceção seja disparada queremos mostrar a mensagem no console, fazendo isso temos o código:

```
using System.ServiceModel;
using Passagens;

class Program
{
    static void Main(string[] args)
    {
        ServiceHost host = new ServiceHost(typeof(ClienteService));

        try
        {
            host.Open();
            Console.ReadLine();
            host.Close();
        }
        catch (Exception ex)
        {
            host.Abort();
            Console.WriteLine(ex.Message);
            Console.ReadLine();
        }
    }
}
```

Agora que temos o código feito, vamos rodar nossa aplicação, para isso vamos setar o projeto **Hosting** como o projeto principal, clicando com o botão direito do mouse sobre o projeto, no *Solution Explorer*, e selecionando *Set as StartUp Project*. Agora podemos clicar em *Start*.

Veja que a aplicação console aparece e uma exceção foi disparada, e ela nos informa que não foi encontrado nenhum *endpoint* no nosso serviço, mas tínhamos definido isso no nosso arquivo de configuração! Porém a exceção nos diz que nenhum arquivo de configuração foi encontrado também. Como podemos resolver isso?

Vamos adicionar um *endpoint* utilizando o método `AddServiceEndpoint`.

```
using System.ServiceModel;
using Passagens;

class Program
{
    static void Main(string[] args)
    {
        ServiceHost host = new ServiceHost(typeof(ClienteService));
        Uri endereco = new Uri("http://localhost:8080/clientes");

        host.AddServiceEndpoint(typeof(IClienteService), new BasicHttpBinding(), endereco);

        try
        {
            host.Open();
            Console.ReadLine();
            host.Close();
        }
        catch (Exception ex)
        {
            host.Abort();
            Console.WriteLine(ex.Message);
            Console.ReadLine();
        }
    }
}
```

```

        host.Abort();
        Console.WriteLine(ex.Message);
        Console.ReadLine();
    }
}

```

E para saber quais *endpoints* estão disponíveis, vamos criar um método que exibe no console todos os *endpoints* do serviço.

```

public static void ExibeInformacoesServico(ServiceHost sh)
{
    Console.WriteLine("{0} online", sh.Description.ServiceType);
    foreach(ServiceEndpoint se in sh.Description.Endpoints)
    {
        Console.WriteLine(se.Address);
    }
}

```

E no `Main` , após abrir o serviço, vamos exibir essas informações:

```

using System.ServiceModel;
using Passagens;

class Program
{
    static void Main(string[] args)
    {
        ServiceHost host = new ServiceHost(typeof(ClienteService));
        Uri endereco = new Uri("http://localhost:8080/clientes");

        host.AddServiceEndpoint(typeof(IClienteService), new BasicHttpBinding(), endereco);

        try
        {
            host.Open();
            ExibeInformacoesServico(host);
            Console.ReadLine();
            host.Close();
        }
        catch (Exception ex)
        {
            host.Abort();
            Console.WriteLine(ex.Message);
            Console.ReadLine();
        }
    }
}

```

ATENÇÃO: Para executar o serviço, o **Visual Studio** precisa estar sendo executado como administrador.

Agora, quando clicamos em *Start*, o nosso serviço roda. Para fins de testes, abra o navegador e digite <http://localhost:8080/clientes> (<http://localhost:8080/clientes>), deve aparecer uma página em branco, porque não fizemos

nenhuma requisição para algum método do serviço. Se digitarmos só <http://localhost:8080> (<http://localhost:8080>), recebemos um erro 404, pois a página não foi encontrada.

O ideal é que exibíssemos as informações do serviço nessa página. Para isso, no *Solution Explorer*, dentro de **Passagens**, vamos clicar com o botão direito do mouse em **App.config** e clicar em *Edit WCF Configuration*. Em *Services -> Passagens.ClienteService -> Host*, vamos alterar o seu endereço base, que é o que exibe as informações do serviço. Clique em *Edit...* e modifique o endereço base para <http://localhost:8080> (<http://localhost:8080>), clique em *OK* e salve o arquivo.

Se clicarmos em *Start* novamente e acessarmos <http://localhost:8080> (<http://localhost:8080>), vemos as informações do serviço.

No próximo capítulo vamos aprender a consumir esse serviço via uma aplicação C#.