WCF Behavior Extensions for Silverlight Consumption

Marcel Veldhuizen

Contents

The archive contains the following files and directories:

MVeldhuizen.DevTools.ServiceBehaviors.dll

Precompiled and signed version of the assembly, containing two WCF service behavior extensions that Silverlight developers might find useful.

The **CrossDomainServiceBehavior** makes it possible to access services hosted by WCF Service Host from Silverlight. Please refer to my blog article for a more detailed description.

The **SilverlightFaultBehavior** changes the HTTP response code sent by WCF when a SOAP fault is sent to the client. This allows Silverlight to handle the error as you're used to from .NET client applications. More information is available in this blog article.

Source

This directory contains the source code to the above assembly, as well as a sample Silverlight application and accompanying WCF service to demonstrate these behaviors. Visual Studio 2010 is required to open the solution file. See further below for a short explanation of the sample application.

Requirements

- Microsoft .NET Framework 3.5 or 4.0
- Visual Studio 2010 to open source code and sample application

Quick Start Guide

The following steps can be followed to apply both behaviors to an existing WCF service.

1. Place the **MVeldhuizen.DevTools.ServiceBehaviors.dll** assembly in the web service's Bin directory, or install it into the global assembly cache (GAC).
2. Declare the behavior extensions in the **App.config** file for the service:

```
<configuration>
  <system.serviceModel>
    <extensions>
      <behaviorExtensions>
        <add name="crossDomainService"
             type="MVeldhuizen.DevTools.ServiceBehaviors.CrossDomainServiceBehavior,
                   MVeldhuizen.DevTools.ServiceBehaviors, Version=1.0.0.0,
                   Culture=neutral, PublicKeyToken=a6b8631d17d033da" />
        <add name="silverlightFaults"
             type="MVeldhuizen.DevTools.ServiceBehaviors.SilverlightFaultBehavior,
                   MVeldhuizen.DevTools.ServiceBehaviors, Version=1.0.0.0,
                   Culture=neutral, PublicKeyToken=a6b8631d17d033da" />
      </behaviorExtensions>
    </extensions>
  </system.serviceModel>
</configuration>
```

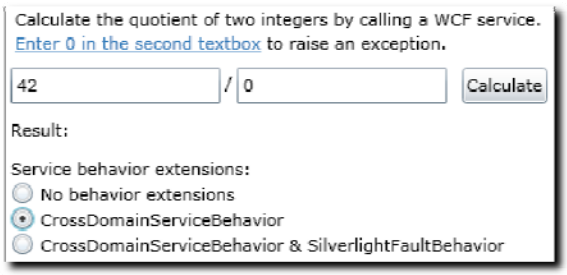
3. Create or modify a service or endpoint behavior:

```
<configuration>
  <system.serviceModel>
    <behaviors>
      <serviceBehaviors>
        <behavior name="myServiceBehavior">
          <serviceMetadata httpGetEnabled="True"/>
          <serviceDebug includeExceptionDetailInFaults="False"/>
          <crossDomainService />
          <silverlightFaults />
        </behavior>
      </serviceBehaviors>
  </system.serviceModel>
</configuration>
```

4. Apply the behavior to your web service:

```
<configuration>
  <system.serviceModel>
    <services>
      <service name="SampleService" behaviorConfiguration="myServiceBehavior">
        <!– Endpoints go here -->
      </service>
  </system.serviceModel>
</configuration>
```

Sample Application



The sample Silverlight application (**SilverlightApp**) should be compiled and run from Visual Studio. It demonstrates the use of both behavior extensions. Three radio buttons can be used to switch between endpoints, each with different server-side configurations:

- Regular basicHttpBinding
- With CrossDomainServiceBehavior
- With CrossDomainServiceBehavior and SilverlightFaultBehavior

The sample application uses a web service to calculate the quotient of two integers. When the divisor is set to 0, a “Division by zero” FaultException will be raised. Depending on which endpoint you have selected, the following will happen:

- Silverlight refuses to call the service, because no clientaccesspolicy.xml can be found.
- The service will function normally, but any FaultException will show up as a CommunicationException as if the service could not be reached.
- “Division by zero” message will appear when a divisor of 0 is used.