WCF Tutorial

Tutorial

+1
<< Previous Next >>

JSON using WCF service

Download Souce:

MyFirstRESTfulService.zip

This article explains about configuring the WCF service to send the response business entity as JSON objects.

JSON –JavaScript Object Notation.

"The JSON text format is syntactically identical to Per-Call Service Per-Session Service Singleton Service JavaScript objects ".

In most of the browser based application, WCF ca g javascript or jquery. When client makes the call to the WCF, JSON or XML is used for mode of communication. WCF has option to send the response in JSON object. This can be configured with ***WebGet*** or ***WebInvoke*** attribute.

In this sample we can create the sample RESTful service to expose the method to read/add/update/delete the employee information. Read the "How to create REST ful Service" articles for more information.

On top of the Restful service we need to update the ResponseMode attribute to send the business entity as JSON object. Below code shows how to configure the JSON response format.

```
[ServiceContract()]
    public interface IEmployeeService
    {
        [WebGet(UriTemplate = "Employee", ResponseFormat=WebMessageFor
        [OperationContract]
        List < Employee > GetAllEmployeeDetails();

        [WebGet(UriTemplate = "Employee?id={id}", ResponseFormat = Web
        [OperationContract]
        Employee GetEmployee(int Id);

        [WebInvoke(Method = "POST", UriTemplate = "EmployeePOST", Resp
        [OperationContract]
        void AddEmployee(Employee newEmp);

        [WebInvoke(Method = "PUT", UriTemplate = "EmployeePUT", Respon
        [OperationContract]
        void UpdateEmployee(Employee newEmp);

        [WebInvoke(Method = "DELETE", UriTemplate = "Employee/{empId}"
        [OperationContract]
        void DeleteEmployee(string empId);
    }
```

You can see that WCF response are send as JSON object while accessing data using browser.

Below sample is the ASP.Net web application is used to explains about the CRUD from WCF service with response as JSON object.

GET Method

```
function RefreshPage() {
         var serviceUrl = "http://saravana:8090/MyService/EmployeeSer

         $.ajax({
             type: "GET",
             url: serviceUrl,
             dataType: 'json',
             contentType: "application/json; charset=utf-8",
             success: function (data) {
                 var itemRow = "< table >";
                 $.each(data, function (index, item) {
                     itemRow += "<tr><td>" + item.EmpId + "</td><td>"
                 });
                 itemRow += "</table>";

                 $("#divItems").html(itemRow);

             },
             error: ServiceFailed
         });
     }
```

POST Method

```
function POSTMethodCall() {
           var EmpUser = [{ "EmpId": "13", "Fname": "WebClientUser", "
           var st = JSON.stringify(EmpUser);
           debugger;
          $.ajax({
              type: "POST",
              url: "http://saravana:8090/MyService/EmployeeService/Emp
              data: JSON.stringify(EmpUser),
               contentType: "application/json; charset=utf-8",
              dataType: "json",
              success: function (data) {
                  // Play with response returned in JSON format
              },
              error:ServiceFailed
          });

      }
<
```

PUT Method

```
function PUTMethodCall() {
         var EmpUser = [{ "EmpId": "3", "Fname": "WebClientUser", "Ln

         $.ajax({
             type: "PUT",
             url: "http://saravana:8090/MyService/EmployeeService/Emp
             data: EmpUser,
             contentType: "application/json; charset=utf-8",
             dataType: "json",
             success: function (data) {
                 // Play with response returned in JSON format
             },
             error: ServiceFailed
         });

     }
<
```

DELETE Method

```
function DELETEMethodCall() {
         $.ajax({
             type: "DELETE",
             url: "http://saravana:8090/MyService/EmployeeService/Emp
             data: "{}",
             contentType: "application/json; charset=utf-8",
             dataType: "json",
             success: function (data) {
                 // Play with response returned in JSON format
             },
             error: function (msg) {
                 alert(msg);
             }
         });

     }
```


<< Previous

Next >>

Tips!

- Always create the service with Interface->Implementation format, mention the contract in Interface.
- Define the service in Class library and refer the class library in Host project. Donâ€™t use service class in host project.
- Change the instance mode to per call as default.
- Always catch exception using try/catch block and throw exception using FaultException < T >.
- Logging and Include exception should be enable while compiling the project in debug mode. While in production deployment disable the logging and Include exception details.