This is revision 1 of the Confirmit Horizons v2020.4.74 CATI Rest API Tutorial, published in April 2020. The information herein describes Confirmit Horizons CATI Supervisor and its features as of Build nr. 2020.4.74. New features may be introduced into the product after this date. Go to www.confirmit.com or check "News" on the Customer Extranet for the latest updates.

What's New in this Revision?

Note: Only the latest changes to this documentation are listed here. Changes made to earlier revisions are listed in the "Changes to the User Documentation" document which can be downloaded from the Confirmit Extranet.

The following changes have been made in revision 1 of the Confirmit Horizons v2020.4.74 CATI Rest API Tutorial:

- This is the first revision of the document. No changes have been made.

Note: The general layout and language in this document is continually being corrected, adjusted and improved to ensure the user has the best possible source of information. Only NEW information and details of functionality that has changed since the previous issue are listed here - minor corrections to the text and document layout are not listed.

Table of Contents

1 Introduction

The Confirmit CATI REST API lets you integrate with Confirmit CATI paradata with your internal systems. Typical use cases include extracting interviewer session and activity related information from the CATI system to calculate interviewer pay or to feed workforce management solutions.

The following areas of Confirmit CATI can be integrated with your systems:

- Exporting of call history and interviewer break data
- Interviewer management
- Assignment management
- Ability to open and close surveys
- Interviewer session data (login and logout timing info)
- Ability to apply CATI survey settings (e.g. applied scheduling script or extended status group)

2 Getting started

The Confirmit Cati REST API is a pure REST/JSON API. Please note that it is not a WCF service, so there is no WSDL available. The API ships with an integration SDK. The SDK is a .NET assembly. If you use Java or JavaScript you cannot use the SDK currently. The SDK contains a set of service and model classes to simplify integration. The Confirmit CATI REST API is an add-on, if you do not have this add-on, you will receive a 503 (Service Unavailable) error when attempting to use it.

How to use an SDK Nuget package:

- Contact Confirmit Support to get CATI Rest SDK;
- Save package in a local Nuget feed;
- Add SDK to the project by right-clicking the project and then choosing Manage Nuget Packages. Search for Confirmit.Cati.Rest.Sdk and click Install.

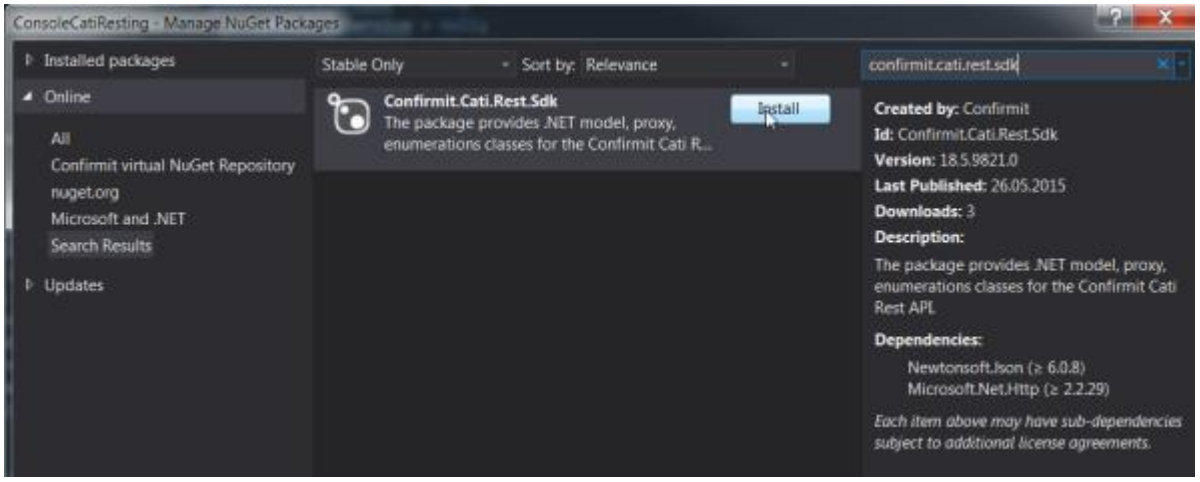

Figure 1 Adding the CATI Rest SDK to a project

Now you can start using the SDK in your application.

3 Using the API SDK

The SDK is a single .NET assembly Confirmit.CATI.REST.SDK. The assembly compiled for the .NET 4.5 framework. SDK contains service and model classes. Service classes act as proxy classes, model classes are mainly POCO classes used in the JSON serialization/deserialization. Service classes are located in the ‘’Confirmit.CATI.REST.SDK.Client” namespace. Here is a list of the service classes:

- BreakHistoryService
- CallHistoryService
- CallHistoryWithVariablesService
- SurveyService
- InterviewerService
- InterviewerSessionHistoryService
- GroupService

All service classes implement corresponding interfaces. Service classes gets single parameter in the constructor – IRestClient. All service methods are asynchronous and returns Task object. In .NET 4.5 you can use the await keyword to wait for the task to complete. Please refer to study code examples.

3.1 Security

To access the API you first need to have xConfirmitApiKey. To get the key you need to call Confirmit LogOn ASMX web service. In the Confirmit.CATI.REST.SDK.LogOn namespace there is a proxy class that can be used to access the LogOn web service.

3.2 Permissions

To be able to call the API the user needs to have SYSTEM_API_ACCESS and SYSTEM_CATI_ADMINISTRATE permissions. Otherwise, a 403 Forbidden error code will be returned.

3.3 Confirmit environment configuration

We suggest using Nordic site to test the Rest API .

- **cati.nordic.confirmit.com** – for the RestClient constructor address parameter;
- **ws.nordic.confirmit.com** – to configure LogOnSoapClient (please see the sample in API SDK)

To access the Confirmit SaaS environment please use the following addresses:

- **cati.euro.confirmit.com** or **cati.us.confirmit.com** – for the RestClient constructor address parameter
- **ws.confirmit.com** or **ws.euro.confirmit.com** – to configure LogOnSoapClient (please see the sample in Using the API SDK)

Site info is published here: https://extranet.confirmit.com/library/saas-info.aspx

You can also use the Environments class to create LogOnSoapClient and RestClient instances for different Confirmit environments without specifying the URLs manually.

3.4 ODATA support

All services have a GetAsync method that returns a list of corresponding records (list of surveys, interviewers, groups, call history entries, break history entries). GetAsync methods are gateways to the OData endpoints. GetAsync gets a single parameter – OData query string. Using a query string, you can filter the result lists. The API supports the OData V4 specification. To filter the result set by the column – use the column name (entity field

name) in the filter. For the sample, see http://www.asp.net/web-api/overview/odata-support-in-aspnet-web-api/supporting-odata-query-options#examples.

The only exception is CallHistoryWithVariablesService which does not use OData query.

3.5 Paging

By default OData endpoints return just the first page of the result set. The page size is 10000 records. To get next pages use the $skip odata option.

3.6 Usage Examples

The samples below demonstrate the following usage examples:

Example 1. Reading call history information without filters

Example 2. Reading call history information from Nordic site with filters

Example 3. Reading interviewer session history information from Nordic site with filter by login time

Example 4. Creating interviewer and interviewer group, assigning on survey

Example 5. Updating interviewer properties

Example 1. Reading call history information without filters

```
// Read configuration parameters
var catiServerAddress = ConfigurationManager.AppSettings.Get("CatiServerAddress");
var wsServerAddress = ConfigurationManager.AppSettings.Get("WsServerAddress");
var proxyServerAddress =
ConfigurationManager.AppSettings.Get("ProxyServerAddress");
var companyId = int.Parse(ConfigurationManager.AppSettings.Get("CompanyId"));
var user = ConfigurationManager.AppSettings.Get("UserName");
var password = ConfigurationManager.AppSettings.Get("Password");
// Create and configure LogOn service proxy class
var logonClient = new LogOnSoapClient();
logonClient.Endpoint.Address = new
EndpointAddress(logonClient.Endpoint.Address.ToString().Replace("localhost",
wsServerAddress));
// Log on and generate xConfirmitApiKey
var xConfirmitApiKey = logonClient.LogOnUser(user, password);
// Create RestClient class
var client = new RestClient(catiServerAddress, proxyServerAddress,
xConfirmitApiKey, companyId);
// Create service class
ICallHistoryService callHistoryService = new CallHistoryService(client);
// Get all call history data
var data = await callHistoryService.GetAsync("");
```

Example 2. Reading call history information from Nordic site with filters

```
// Read configuration parameters
var companyId = int.Parse(ConfigurationManager.AppSettings.Get("CompanyId"));
var user = ConfigurationManager.AppSettings.Get("UserName");
var password = ConfigurationManager.AppSettings.Get("Password");
// Create and configure LogOn service proxy class
var logonClient = Environments.Nordic.CreateLogOnSoapClient();
// Log on and generate xConfirmitApiKey
var xConfirmitApiKey = logonClient.LogOnUser(user, password);
// Create RestClient class
var client = Environments.Nordic.CreateCatiRestClient(xConfirmitApiKey, companyId);
// Create service class
ICallHistoryService callHistoryService = new CallHistoryService(client);
// Get all call history data
var data = await callHistoryService.GetAsync("$filter=CallCenterId eq 1 and
InterviewId eq 11");
```

Example 3. Reading interviewer session history information from Nordic site with filter by login time

```
// Read configuration parameters
var companyId = int.Parse(ConfigurationManager.AppSettings.Get("CompanyId"));
var user = ConfigurationManager.AppSettings.Get("UserName");
var password = ConfigurationManager.AppSettings.Get("Password");
// Create and configure LogOn service proxy class
var logonClient = Environments.Nordic.CreateLogOnSoapClient();
// Log on and generate xConfirmitApiKey
var xConfirmitApiKey = logonClient.LogOnUser(user, password);
// Create RestClient class
var client = Environments.Nordic.CreateCatiRestClient(xConfirmitApiKey, companyId);
// Create service class
IInterviewerSessionHistoryService interviewerSessionHistoryService = new
InterviewerSessionHistoryService(client);
// Get all call history data
var data = await
interviewerSessionHistoryService.GetAsync("$filter=cast(LoginTime,'Edm.DateTimeOffs
et') gt 2020-01-02T00:00:00");
```

Example 4. Creating interviewer and interviewer group, assigning on survey

```
// Read configuration parameters
var companyId = int.Parse(ConfigurationManager.AppSettings.Get("CompanyId"));
var user = ConfigurationManager.AppSettings.Get("UserName");
var password = ConfigurationManager.AppSettings.Get("Password");
// Create and configure LogOn service proxy class
var logonClient = Environments.Nordic.CreateLogOnSoapClient();
// Log on and generate xConfirmitApiKey
var xConfirmitApiKey = logonClient.LogOnUser(user, password);
// Create RestClient class
var client = Environments.Nordic.CreateCatiRestClient(xConfirmitApiKey, companyId);
// Create service classes
var interviewerService = new InterviewerService(client);
var groupService = new GroupService(client);
var surveyService = new SurveyService(client);
// Create new interviewer group
var group = new Group
{
    Name = "TestGroup_" + Guid.NewGuid()
};
var groupId = await groupService.Create(group);
// Create new interviewer in the new group
var interviewerProperties = new InterviewerProperties
{
    Name = "TestInter_" + Guid.NewGuid(),
    Location = "Foo",
    Password = "123",
    AssignmentsListMode = AssignmentListMode.AllCalls
};
interviewerProperties.ParentGroups.Add(groupId);
var interviewerId = await interviewerService.Create(interviewerProperties);
// Assign interviewer on the survey
var surveys = await surveyService.GetAsync("");
var survey = surveys.FirstOrDefault();
await interviewerService.AssignOnSurvey(interviewerId, survey.SurveyId);
```

Example 5. Updating interviewer properties

```
// Read configuration parameters
var companyId = int.Parse(ConfigurationManager.AppSettings.Get("CompanyId"));
var user = ConfigurationManager.AppSettings.Get("UserName");
var password = ConfigurationManager.AppSettings.Get("Password");
// Create LogOn service class, log on and generate xConfirmitApiKey
var logonClient = Environments.Nordic.CreateLogOnSoapClient();
var xConfirmitApiKey = logonClient.LogOnUser(user, password);
// Create RestClient class
var client = Environments.Nordic.CreateCatiRestClient(xConfirmitApiKey, companyId);
// Create service classes
var interviewerService = new InterviewerService(client);
var surveyService = new SurveyService(client);
// Create new interviewer in the new group
var interviewerProperties = new InterviewerProperties
{
    Name = "TestInter_" + Guid.NewGuid(), Location = "Foo", Password = "123"
};
var interviewerId = await interviewerService.Create(interviewerProperties);
// Change interviewer properties
interviewerProperties.InterviewerId = interviewerId;
interviewerProperties.Name = "TestInterNew_" + Guid.NewGuid();
interviewerProperties.Location = "FooNew";
interviewerProperties.Password = "123New";
interviewerProperties.Description = "Test interviewer";
interviewerProperties.Mode = TaskChoiceMode.Manual;
interviewerProperties.AssignmentsListMode = AssignmentListMode.AssignedCallsOnly;
interviewerProperties.AllowedTaskChoice = new List<TaskChoicePermissions>
{
    TaskChoicePermissions.Automatic,
    TaskChoicePermissions.Manual,
    TaskChoicePermissions.SurveyAssignment
};
string surveyId = (await surveyService.GetAsync("")).First().SurveyId;
interviewerProperties.AutomaticSurveyId = surveyId;
interviewerProperties.CallGroupId = 0;
interviewerProperties.CallCenterId = 1;
interviewerProperties.DialType = DialType.Manual;
// Update interviewer properties
await interviewerService.Update(interviewerProperties);
```