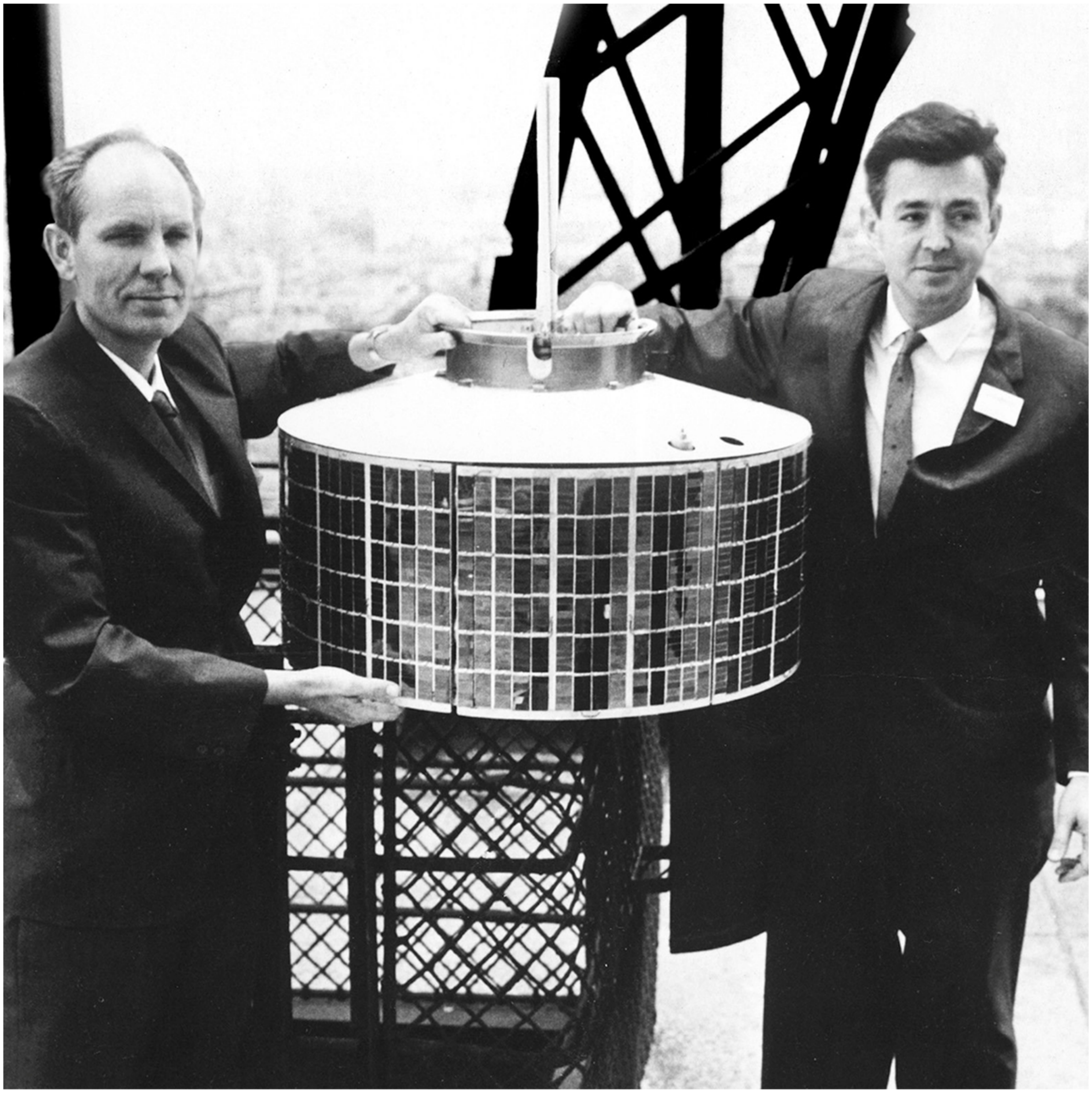
netTerrain 10.1

SOAP API Programming Guide



Contents

Document Code. **GN_D_n10-05**
Last revision: **02/20/2026**

Image: **Harold Rosen** (right) (20 March 1926 – 30 January 2017).
Harold Rosen was an American electrical engineer, known as "the father of the geostationary satellite". The team that he built and led at the Hughes Aircraft Company designed the first geosynchronous communications satellite, Syncom.

Graphical Networks LLC
Telephone: +1-240-912-6223
Fax: +1-240 something something

1 About this guide

1.1 Who should use it

This guide is for all those software developers out there that can't wait to create integrations with netTerrain. It's also dedicated to those netTerrain power users that want to add their special foo and are too anxious to wait for that pet feature in the roadmap. Yes, you can create your own custom menus in netTerrain and train them to do a bunch of things.

Users of the API are typically programmers familiar with C# (or Java or Python or whatever you get your hands on that can connect to WCF). If you are not familiar with any of these vastly superior OO languages and all you did was write some spaghetti PHP scripts or hack some SQL you are still welcome, we forgive you. Plenty of examples are provided throughout the guide to help you along the way.

Now go turn those cups of coffee into lines of code.

1.2 SOAP vs REST API

This is our SOAP Programming guide. We also have a super robust REST API. There are a few instances where the SOAP API is a better choice:

1) you need to create a custom dll that you want to launch from netTerrain as a node double-click

2) You want to create an event trigger (such as an on-property change trigger) that is executed on the server when a property value for a given node changes

3) You want to create a custom action using the context menu

The three instances mentioned above are examples where you probably want to use the netTerrain SOAP API to interact with netTerrain. In section 2.5 we'll go over these three examples. In most other cases you are probably better off using the REST API. To use the REST API simply refer to the Swagger documentation directly available online on the help ribbon of your netTerrain application.

1.3 Assumptions

The Web API description guide assumes you have a good understanding of the following concepts:

- netTerrain (that thing that Graphical Networks makes)
- The difference between an instance and a type (in netTerrain terms). Ok, in netTerrain a type is defined in the catalog (chair), and an instance is a manifestation of that type in the project (the second chair on the left, in the blue conference room). It is akin to class vs instantiation of that class in OO programming. Now you know.
- Node and link properties and their values
- Object oriented programming languages (using C# for example)

1.4 Guide format

Throughout the guide, we use C# as our reference language. This does not mean that the netTerrain API is language specific. You can write external applications that interact with netTerrain using another programming language like Java. If your framework can reference a wsdl file and use SOAP envelopes to connect to WCF you are good. Consider that the methods for gaining access to the API, the syntax and examples may differ for other platforms or languages.

In the API reference guide, we review each method that is exposed through the API interface including the syntax, exceptions, remarks and an example.

For the method syntax we use C# as our reference language and review each input parameter associated with the method signature.

In the exceptions paragraph we specify the exception notation and condition for an exception to be raised.

In the remarks section we may review special aspects to consider, such as enumerations, default values, return values and so on.

Finally, every method includes one example, written in C#.

2 SOAP API Basics

The netTerrain Application Programming Interface (API) is based on the Microsoft Windows Communication Foundation (WCF) framework and it is implemented as a web-accessible SSL-protected WCF service.

Some use cases for the netTerrain API include:

- Inserting, updating or deleting objects in netTerrain from an external application
- Extracting information from netTerrain programmatically for reporting or analysis purposes
- Synchronizing netTerrain with other application data in real-time
- Creating your own special menus or custom double click behaviors

Users familiar with the netTerrain Integration Toolkit (ITK) and the netTerrain database may achieve similar results using the ITK or tapping into the netTerrain backend database. There are certain advantages to using the API though:

- It is asynchronous in nature: as opposed to the ITK, which polls data from a source in a synchronous fashion or upon manual triggering, you can push data asynchronously into netTerrain. This could be a better alternative for certain real-time applications, or to reduce the number of processes used to update data in netTerrain
- It is database structure independent: using the API for data extraction or reporting purposes is independent of any changes that happen to the netTerrain backend database structure.

This guide provides step–by–step instructions on how to set-up and use the netTerrain API with a deployed netTerrain instance (or better). It includes descriptions of all the API methods along with some examples.

All netTerrain API methods obey application layer business rules and work in the same way as the equivalent operations performed through the graphical user interface.

2.1 Setting up access to the API

To make the netTerrain API accessible to an external application it is necessary to first make some modifications to the web.config file.

Open the web.config file in a text editor and insert the following code inside the tag:

```
  <system.serviceModel>
     <services>
         <service name="NetTerrain.WebApi.WebApiService">
             <endpoint address="" binding="wsHttpBinding"
bindingConfiguration="WebApiSecureBinding" behaviorConfiguration=""
contract="NetTerrain.WebApi.INetTerrainWebApi" />
             <endpoint address="mex" binding="mexHttpsBinding"
contract="IMetadataExchange" />
         </service>
     </services>
```

```
    <behaviors>
        <serviceBehaviors>
            <behavior name="">
                <serviceMetadata httpGetEnabled="false"
httpsGetEnabled="true" />
                <serviceDebug includeExceptionDetailInFaults="true" />
                <serviceCredentials>
                    <serviceCertificate
findValue="eng.graphicalnetworks.com"
     storeLocation="LocalMachine" storeName="My"
x509FindType="FindBySubjectName" />
                    <userNameAuthentication
userNamePasswordValidationMode="MembershipProvider"
membershipProviderName="NetTerrainMembershipProvider" />
                    <clientCertificate>
                        <authentication
certificateValidationMode="None" />
                    </clientCertificate>
                </serviceCredentials>
            </behavior>
        </serviceBehaviors>
    </behaviors>
    <bindings>
        <wsHttpBinding>
            <binding name="WebApiSecureBinding">
                <security mode="TransportWithMessageCredential">
                    <transport clientCredentialType="None"
proxyCredentialType="None" realm="" />
                    <message clientCredentialType="UserName"
algorithmSuite="Default" />
                </security>
            </binding>
        </wsHttpBinding>
    </bindings>
    <serviceHostingEnvironment multipleSiteBindingsEnabled="true" />
</system.serviceModel>
```

We will briefly review some important aspects of the WCF service configuration tags in your web.config file, necessary for netTerrain to properly enable API access. Please refer to the Microsoft WCF guide for a more comprehensive description of these WCF service settings.

2.1.1 WCF Service and endpoint address

The netTerrain API is declared as a WCF service named NetTerrain.WebApi.WebApiService. A client application uses this name to identify the service. Services are described in terms of endpoints.

```
<endpoint address=""
    binding="wsHttpBinding"
    bindingConfiguration="WebApiSecureBinding"
    behaviorConfiguration=""
    contract="NetTerrain.WebApi.INetTerrainWebApi"/>
```

The service endpoint describes how a client interacts with the service. The endpoint combines the address, binding and contract settings of a service. The table below provides more insights into how the WCF service is comprised.

Attribute Name	Value	Description
address		Specifies where to send messages for this endpoint. This setting may not be required as the API uses just one default endpoint.
binding	wsHttpBinding	Specifies how to send a message for this endpoint. wsHttpBinding is a protocol providing enough security for the purposes of the API.
bindingConfiguration	WebApiSecureBinding	A reference to the binding configuration in the bindings section (described later).
behaviorConfiguration		A reference to the behavior configuration in the behavior section. Usually no value needs ito be provided as only one default behavior is used.
contract	NetTerrain.WebApi.INetTerrainWebApi	Specifies the content of messages for this endpoint. This is, in essence, a list of API methods.

Clients use a standard endpoint to receive service metadata. This endpoint also specifies the SSL security settings that will be used for the exchange metadata.

The Endpoint address tag is set up as follows:

```
<endpoint address="mex" binding="mexHttpsBinding"
    contract="IMetadataExchange">
```

2.1.2 Behavior

Let’s review some of the behavior related tags in your WCF configuration. For starters, the serviceMetadata should be configured as follow:

```
<serviceMetadata httpGetEnabled="false" httpsGetEnabled="true"/>
```

This tag includes directives to prevent service metadata extraction by a client through the http protocol, and instead, enables the https protocol in order to operate in a similar secured environment as normal browser-based end users do.

Next, the serviceDebug tag provides a flag to obtain exception data in case of faults during the execution of a service method:

```
<serviceDebug includeExceptionDetailInFaults="true"/>
```

The service credentials section sets up the server certificate data for working through SSL. For security reasons, the API should only work with your netTerrain instance by means of a signed X509 certificate:

```
<serviceCertificate findValue="eng.graphicalnetworks.com"
    storeLocation="LocalMachine"
    storeName="My"
    x509FindType="FindBySubjectName"/>
```

Notice the locator that describes where to find the certificate. In our example the certificate is searched by its name, such as "eng.graphicalnetworks.com".

As mentioned before, we want the API user to be authenticated in a similar way as through the regular browser interface. To achieve that, we use the userNameAuthentication tag, which sets up the mechanism by which credentials are passed. We highly recommend keeping this section the same was as shown below:

```
<userNameAuthentication
    userNamePasswordValidationMode="MembershipProvider"
    membershipProviderName="NetTerrainMembershipProvider"/>
```

Finally, the clienCertificate tag shown below disables the requirement for an installed certificate on the client machine. Otherwise, an API client would also need the certificate on their machine.

```
<clientCertificate>
    <authentication certificateValidationMode="None"/>
</clientCertificate>
```

2.1.3 Binding

A binding protocol is used by the service to control the client/server interaction. The Web API uses the so-called wsHttpBinding protocol for that purpose. The binding section contains just one configuration parameter (WebApiSecureBinding) and the main service endpoint refers to it. It specifies that the Web API user will need to send his netTerrain credentials to use the service. The credentials validation process occurs on every message transmission (essentially every method call) while the client/server interaction uses Transport Layer Security.

```
<security mode="TransportWithMessageCredential">
    <transport clientCredentialType="None" proxyCredentialType="None"
realm=""/>
    <message clientCredentialType="UserName"
algorithmSuite="Default" />
</security>
```

2.2 Checking API accessibility

Your netTerrain application includes a special file used by the API called NetTerrainWebApi.svc, which is located in the WebApi folder under the netTerrain installed path:

```
c:\(netTerrain path)\WebApi\NetTerrainWebApi.svc.
```

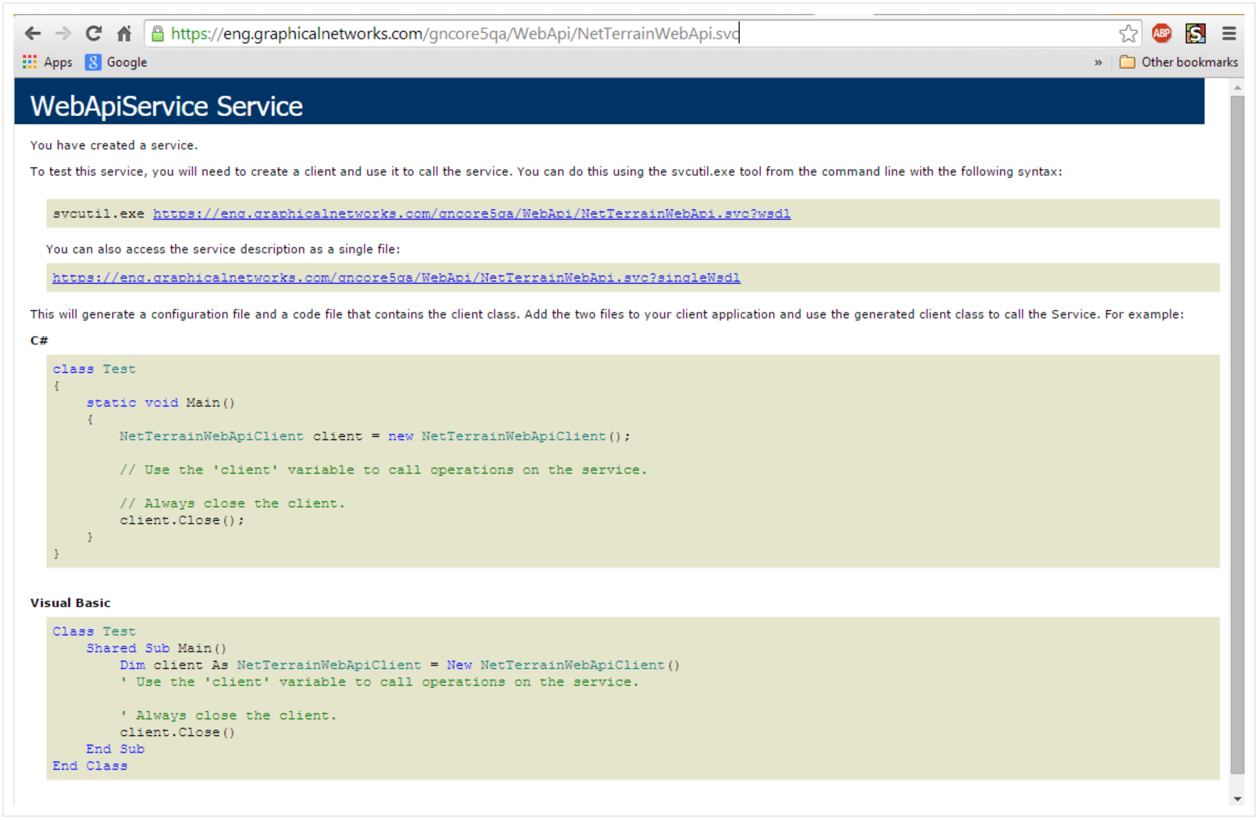
This file contains a @ServiceHost WCF directive intended to host the Web API service:

```
<%@ ServiceHost Language="C#" Debug="true"
Service="NetTerrain.WebApi.WebApiService" %>
```

After the API settings are added to the web.config file you can proceed to access the NetTerrainWebApi.svc file from a web browser by its URL address, as we show in the example below:

```
https://(netTerrainURL)/WebApi/NetTerrainWebApi.svc
```

This will force the directive to run and if the service creation process was successful, the following dialog should appear:


https://eng.graphicalnetworks.com/gncore5qa/WebApi/NetTerrainWebApi.svc
Apps
Google
Other bookmarks

WebApiService Service

You have created a service.

To test this service, you will need to create a client and use it to call the service. You can do this using the svcutil.exe tool from the command line with the following syntax:

```
svcutil.exe https://eng.graphicalnetworks.com/gncore5qa/WebApi/NetTerrainWebApi.svc?wsdl
```

You can also access the service description as a single file:

```
https://eng.graphicalnetworks.com/gncore5qa/WebApi/NetTerrainWebApi.svc?singleWsdl
```

This will generate a configuration file and a code file that contains the client class. Add the two files to your client application and use the generated client class to call the Service. For example:

C#

```
class Test
{
    static void Main()
    {
        NetTerrainWebApiClient client = new NetTerrainWebApiClient();

        // Use the 'client' variable to call operations on the service.

        // Always close the client.
        client.Close();
    }
}
```

Visual Basic

```
Class Test
    Shared Sub Main()
        Dim client As NetTerrainWebApiClient = New NetTerrainWebApiClient()
        ' Use the 'client' variable to call operations on the service.

        ' Always close the client.
        client.Close()
    End Sub
End Class
```

Next, the Web API service is created, and you should be able to start working with the netTerrain API.

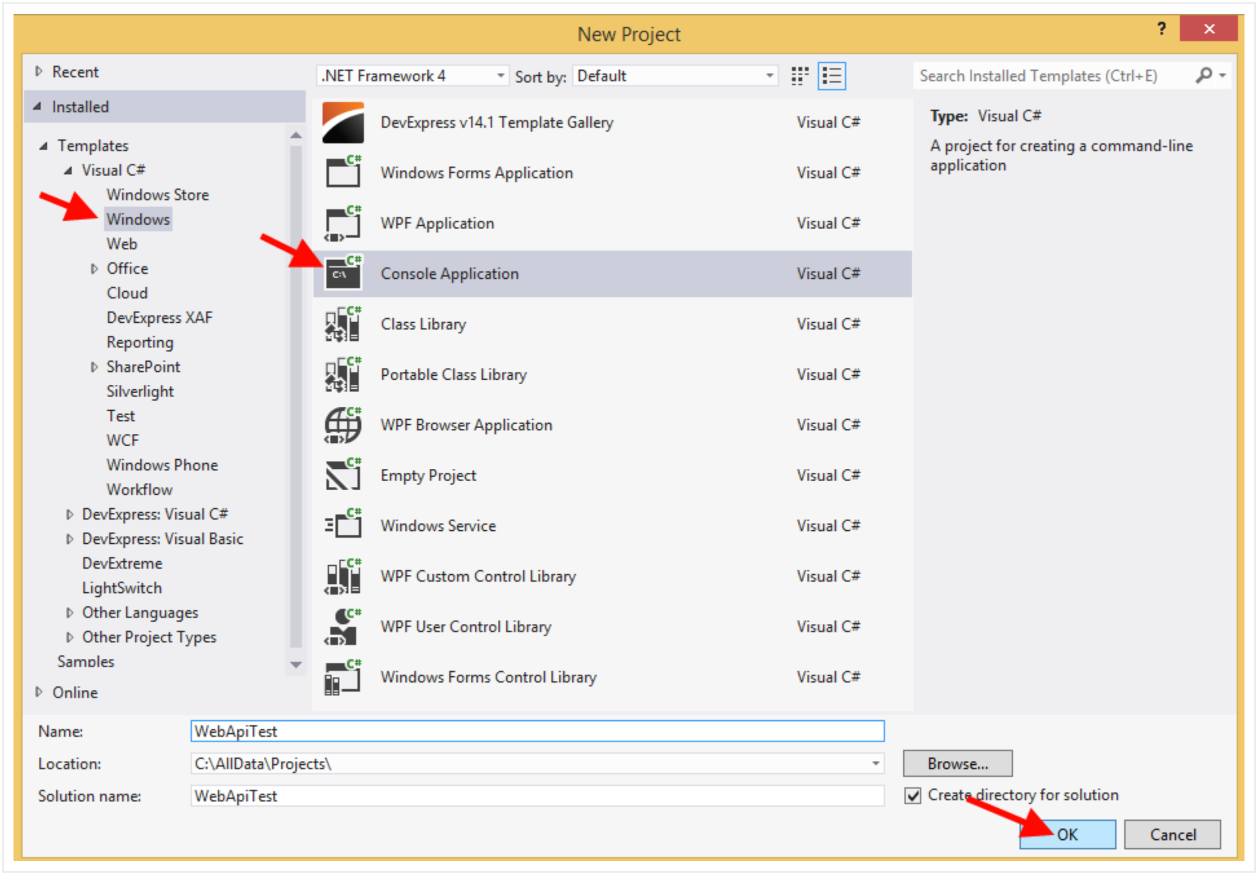
2.3 Using the API

As mentioned before, the netTerrain API is not language specific. You can write external applications that interact with netTerrain using a language like Java. Take into account though, that the methods for gaining access to the API may differ for other platforms or languages.

In order to demonstrate a client-side application using the netTerrain Web API, a simple C# console application will be created using Microsoft Visual Studio 2012. It is recommended to install version 4.0 of the Microsoft .Net Framework on your client machine.

2.3.1 Creating a test console application

To start the creation of our simple test console application we first open Microsoft Visual Studio and create a Visual C# console application:



Type: Visual C#

A project for creating a command-line application

Name: WebApiTest

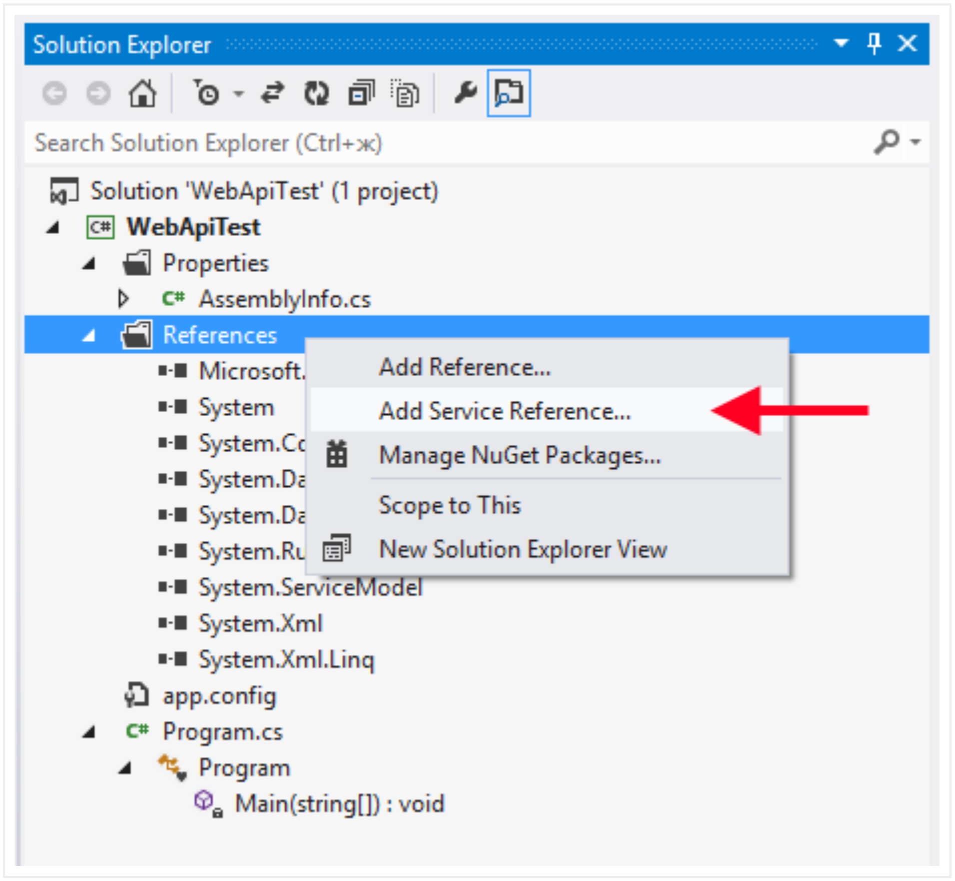
Location: C:\AllData\Projects\

Solution name: WebApiTest

[✓] Create directory for solution

To make the Web API methods available by the application, a service reference should be added:

1. Select the 'Add Service Reference' menu item after right clicking on the References folder of your application:



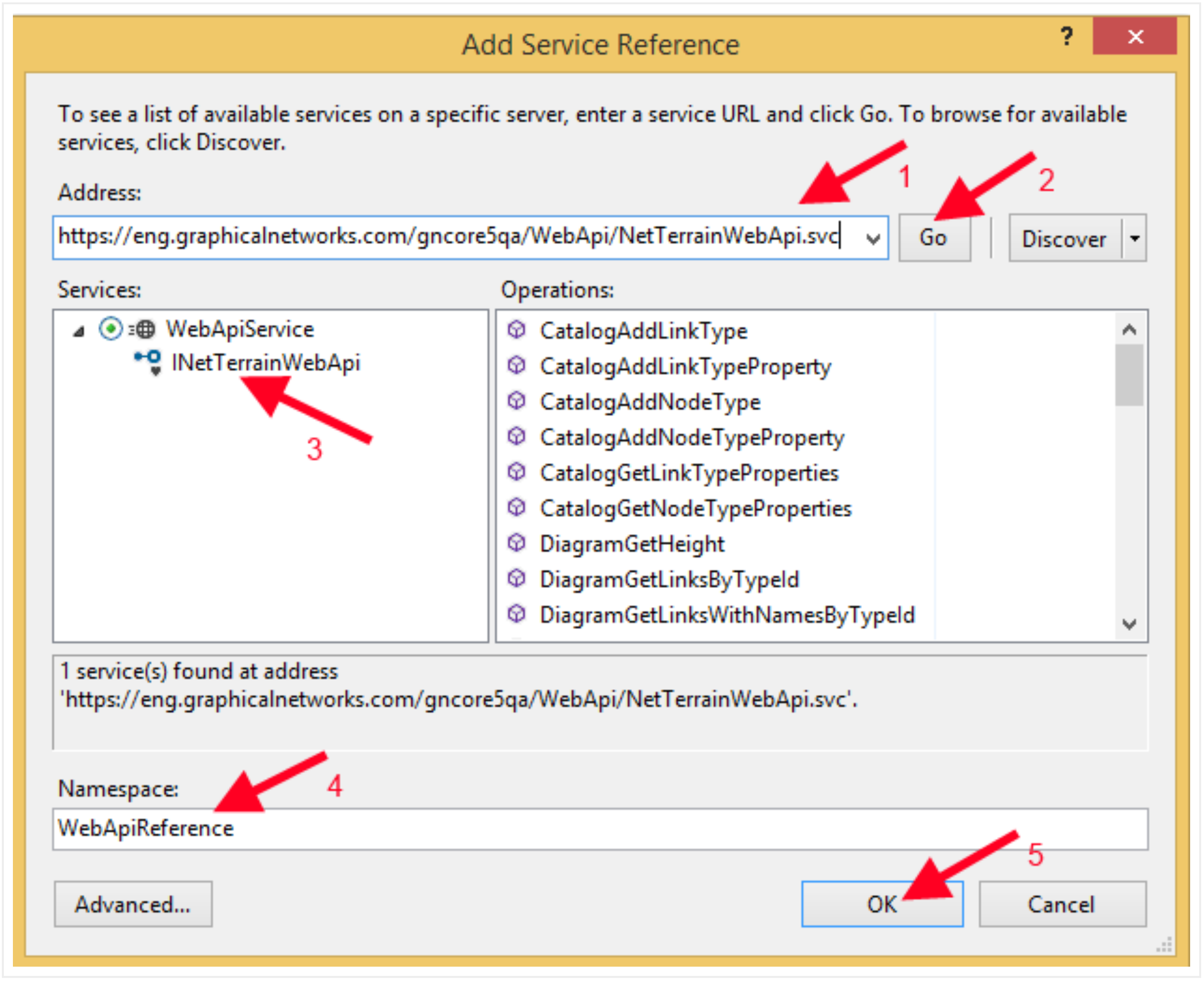
2) Type the full Web API URL address into the address field.

3) Click on ‘Go’. If the WebApiService service is accessible, it will appear in the window below.

4) Expand it to ensure it contains the INetTerrainWebApi interface and review the methods on the right.

5) Specify the service reference name you want to use in your application.

6) Click ‘Ok’.



To see a list of available services on a specific server, enter a service URL and click Go. To browse for available services, click Discover.

Address:

https://eng.graphicalnetworks.com/gncore5qa/WebApi/NetTerrainWebApi.svc

Services:

- WebApiService
  - INetTerrainWebApi

Operations:

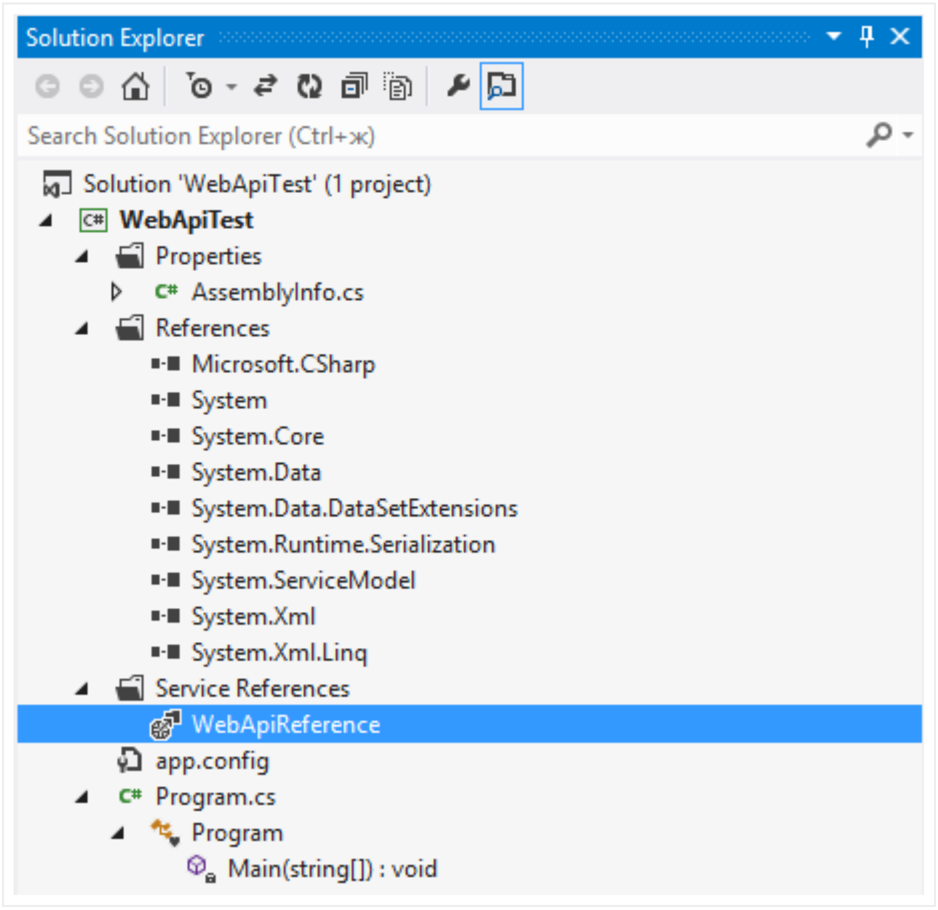
- CatalogAddLinkType
- CatalogAddLinkTypeProperty
- CatalogAddNodeType
- CatalogAddNodeTypeProperty
- CatalogGetLinkTypeProperties
- CatalogGetNodeTypeProperties
- DiagramGetHeight
- DiagramGetLinksByTypeId
- DiagramGetLinksWithNamesByTypeId

1 service(s) found at address 'https://eng.graphicalnetworks.com/gncore5qa/WebApi/NetTerrainWebApi.svc'.

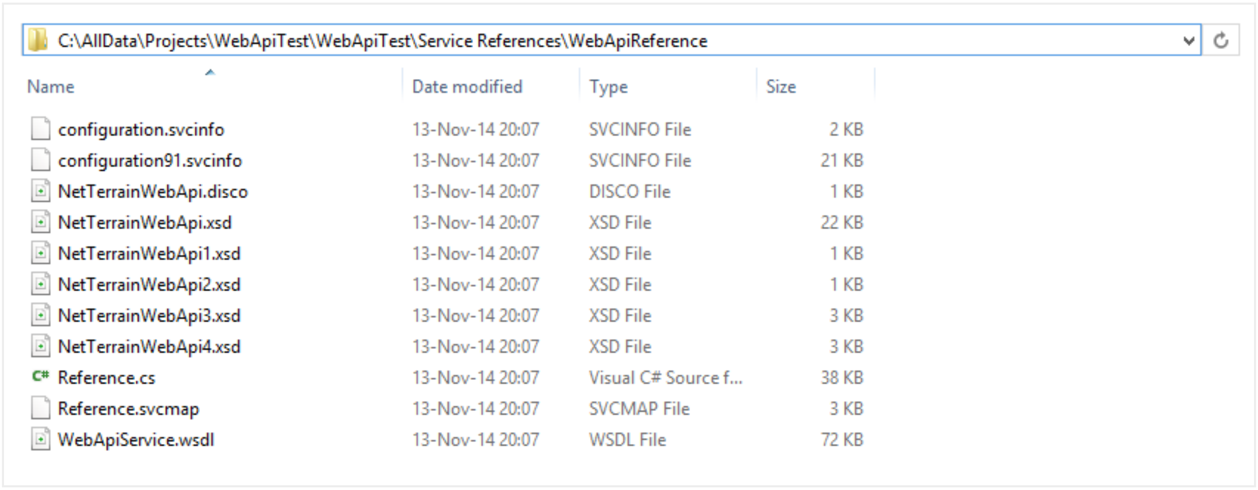
Namespace:

WebApiReference

7) The reference to the Web API will appear in your Solution Explorer:



Visual Studio will create a WebApiReference folder and a set of files implementing the service interface to make its methods accessible from your code:



C:\AllData\Projects\WebApiTest\WebApiTest\Service References\WebApiReference

Name	Date modified	Type	Size
configuration.svcinfo	13-Nov-14 20:07	SVCINFO File	2 KB
configuration91.svcinfo	13-Nov-14 20:07	SVCINFO File	21 KB
NetTerrainWebApi.disco	13-Nov-14 20:07	DISCO File	1 KB
NetTerrainWebApi.xsd	13-Nov-14 20:07	XSD File	22 KB
NetTerrainWebApi1.xsd	13-Nov-14 20:07	XSD File	1 KB
NetTerrainWebApi2.xsd	13-Nov-14 20:07	XSD File	1 KB
NetTerrainWebApi3.xsd	13-Nov-14 20:07	XSD File	3 KB
NetTerrainWebApi4.xsd	13-Nov-14 20:07	XSD File	3 KB
Reference.cs	13-Nov-14 20:07	Visual C# Source f...	38 KB
Reference.svcmap	13-Nov-14 20:07	SVCMAP File	3 KB
WebApiService.wsdl	13-Nov-14 20:07	WSDL File	72 KB

8) To ensure that the Web API interface is visible declare it in the program. At this stage the whole main program should look like this:

```
namespace WebApiTest
{
  class Program
  {
    static void Main(string[] args)
    {
      WebApiReference.INetTerrainWebApi connection = null;
    }
  }
}
```

Ensure that the WebApiReference namespace is visible and build the application. The code should compile without any errors.

2.3.2 Adding the connection factory

As a next step, and in preparation for some netTerrain specific code, we will add a private method in our Program class encapsulating the connection settings. In general, these settings merge the ones corresponding to the web.config file described previously.

As a rule of thumb, the method should work as if no settings in your web.config file were modified. To make the connection factory work correctly in terms of the credentials, add the following parameters:

Name	Type	Description
url	string	An address to the web service, which can point to another instance of netTerrain.
username	string	The netTerrain user name to obtain permissions. Currently all methods are exposed to any user, as such only admin type user connections are supported
password	string	

Let's take a closer look at our method source code:

```
 private static ChannelFactory<WebApiReference.INetTerrainWebApi>
CreateConnectionFactory(
 string url, string username, string password)
 {
    // 1\. Set up binding options: TransportWithMessageCredential
mode:
    // Transport forces the use of the HTTPS protocol.
    // MessageCredential specifies that the WCF service will perform
credentials validation on
    // every service method call.
    WSHttpBinding binding = new WSHttpBinding();
    binding.Security.Mode =
SecurityMode.TransportWithMessageCredential;
    binding.Security.Message.ClientCredentialType =
MessageCredentialType.UserName;

    // 2\. Endpoint address sets the service Uri.
    EndpointAddress endpointAddress = new EndpointAddress(new
Uri(url));

    // 3\. Channel factory combines binding options with the service
address.
    ChannelFactory<WebApiReference.INetTerrainWebApi> factory =
    new ChannelFactory<WebApiReference.INetTerrainWebApi>(binding,
endpointAddress);

    // 4\. Add user credentials to the factory.
    // The service accepts native netTerrain or Active Directory
users.
    // To distinguish AD users from native ones the AD postfix '|
AD_H4fU7d' must be added:
    factory.Credentials.UserName.UserName = username;
    factory.Credentials.UserName.Password = password;

    // 5\. This line removes client-side certificate validation.
Otherwise we'll need the same certificate
    // installed on the client machine.

factory.Credentials.ServiceCertificate.Authentication.CertificateValidati
=
    X509CertificateValidationMode.None;

    return factory;
 }
```

This method requires the following references that must be added at the beginning of your code (before the namespace definition):

```
using System;
using System.ServiceModel;
using System.ServiceModel.Security;
```

Your application should now be able to build, without any errors.

2.3.3 Creating and testing the web service connection

With the helper method just added, it is now very easy to create the connection:

```
 ChannelFactory<WebApiReference.INetTerrainWebApi> factory =
CreateConnectionFactory(
  @"https://eng.graphicalnetworks.com/gncore5qa/WebApi/
NetTerrainWebApi.svc",
  "username",
  "password");
 WebApiReference.INetTerrainWebApi connection =
factory.CreateChannel();
```

If you are passing credentials as an Active Directory user, refer to the syntax bellow:

```
 ChannelFactory<WebApiReference.INetTerrainWebApi> factory =
CreateConnectionFactory(
  @"https://eng.graphicalnetworks.com/gncore5qa/WebApi/
NetTerrainWebApi.svc",
  "username|AD_H4fU7d", // Username postfix to distinguish AD users.
  "password");
 WebApiReference.INetTerrainWebApi connection =
factory.CreateChannel();
```

Now that we have a channel for the factory, we can proceed to test it. The following syntax is not only intended to test the connection itself, but can also precede every method you write:

```
 try
 {
    // Test call.
    connection.TestConnection();
 }
 catch (FaultException<FaultInfo> info)
 {
    // FaultException is the type that refers to any errors caught via
Web API.
    .......// The "info" parameter may contain
    // some useful data about the cause of the error.
    Console.WriteLine(string.Format("Connection failed: {0}",
info.Detail.Details));
 }
 catch (Exception ex)
 {
    // Unknown exception.
    Console.WriteLine(string.Format("Connection failed: {0}",
ex.Message));
 }
 Console.WriteLine("Connection established.");
 Console.ReadKey();
```

Our TestConnection sample code above does nothing except check if a connection was established correctly. In case of problems an exception will be raised. Note that the WCF specification uses Faults rather than Exceptions, to pass useful information about errors through the open channel. As such, the following reference is required at the beginning of your code to support the FaultInfo class:

```
using WebApiTest.WebApiReference;
```

Below is the complete code for the Main method (without references):

```
static void Main(string[] args)
 {
    // Create connection.
    ChannelFactory<WebApiReference.INetTerrainWebApi> factory =
CreateConnectionFactory(
    @"https://eng.graphicalnetworks.com/gncore5qa/WebApi/
NetTerrainWebApi.svc",
    "username", // Username postfix to distinguish AD users.
    "password");
    WebApiReference.INetTerrainWebApi connection =
factory.CreateChannel();
  try
```

```
    {
      // Test call.
      connection.TestConnection();
    }
    catch (FaultException<FaultInfo> info)
    {
      // FaultException is the type that refers to any errors caught via
  Web API.
      .......// The "info" parameter may contain
      // some useful data about the cause of the error.
      Console.WriteLine(string.Format("Connection failed: {0}",
  info.Detail.Details));
    }
    catch (Exception ex)
    {
      // Unknown exception.
      Console.WriteLine(string.Format("Connection failed: {0}",
  ex.Message));
    }
    Console.WriteLine("Connection established.");
    Console.ReadKey();
   }
```

2.4 Setting up your Web API with no SSL protection

You can configure your Web API to interface netTerrain via an unprotected communications channel. The recommended setting to use this unsecured configuration is a controlled environment (such as a secure LAN network) where the traffic between the client and the service cannot be intercepted from the outside.

2.4.1 Configuring the web.config file for unprotected access

The full configuration is shown here:

```
<system.serviceModel>
    <services>
        <service name="NetTerrain.WebApi.WebApiService">
            <endpoint address="" binding="customBinding"
bindingConfiguration="AllowInsecureTransportBinding"
                contract="NetTerrain.WebApi.INetTerrainWebApi" />
            <endpoint address="mex" binding="mexHttpBinding"
```

```
contract="IMetadataExchange" />
        </service>
    </services>
    <behaviors>
        <serviceBehaviors>
            <behavior name="">
                <serviceMetadata httpGetEnabled="true" />
                <serviceDebug includeExceptionDetailInFaults="true" />
                <serviceCredentials>
                    <userNameAuthentication
userNamePasswordValidationMode="MembershipProvider"

membershipProviderName="NetTerrainMembershipProvider" />
                </serviceCredentials>
            </behavior>
        </serviceBehaviors>
    </behaviors>
    <bindings>
        <customBinding>
            <binding name="AllowInsecureTransportBinding">
                <textMessageEncoding />
                <security authenticationMode="UserNameOverTransport"
   allowInsecureTransport="true" />
                <httpTransport />
            </binding>
        </customBinding>
    </bindings>
</system.serviceModel>
```

2.4.2 Configuring the client application for unprotected access

To allow an end-user client application to access an unprotected Web API channel factory, the source code will require changes to match the corresponding settings of your configuration file:

```
 private static ChannelFactory<WebApiReference.INetTerrainWebApi>
CreateConnectionFactory(
 string url, string username, string password)
 {
    // 1\. Configure security properties for unprotected access.
    var security =
SecurityBindingElement.CreateUserNameOverTransportBindingElement();
```

```
    security.EnableUnsecuredResponse = true;
    security.AllowInsecureTransport = true;

    // 2\. Create custom binding element with unprotected security,
text message encoding
    // and http transport.
    var customBinding = new CustomBinding(security,
    new TextMessageEncodingBindingElement(),
    new HttpTransportBindingElement());

    // 3\. Endpoint address sets the service Uri.
    EndpointAddress endpointAddress = new EndpointAddress(new
Uri(url));

    // 4\. Channel factory combines binding options with the service
address.
    ChannelFactory<WebApiReference.INetTerrainWebApi> factory =
    new
ChannelFactory<WebApiReference.INetTerrainWebApi>(customBinding,
                                                     endpointAddress);

    // 5\. Add user credentials to the factory.
    // The service accepts native netTerrain or Active Directory
users.
    // To distinguish AD users from native ones the AD postfix '|
AD_H4fU7d' must be added
    // to the username.
    factory.Credentials.UserName.UserName = username;
    factory.Credentials.UserName.Password = password;

    return factory;
 }
```

2.5 Custom Modules

The functionality of your netTerrain application can be extended with custom modules (also referred to as "extension modules"). These are user generated dynamic-link libraries (or DLL extensions) written in C# or any other language that supports WCF and Named Pipes (more on that later).

From the netTerrain point of view, a DLL extension is a client application that uses the netTerrain Web API. The difference between a custom module and an external application that uses the netTerrain Web API is that the custom module is integrated into your netTerrain server and accessible from the netTerrain GUI through a context menu or a node double click behavior, whereas the external application accesses netTerrain from the outside, is not exposed in the netTerrain GUI and requires authentication to work. As opposed to a regular netTerrain Web API client application, custom modules require no user interaction to establish a connection to the service.

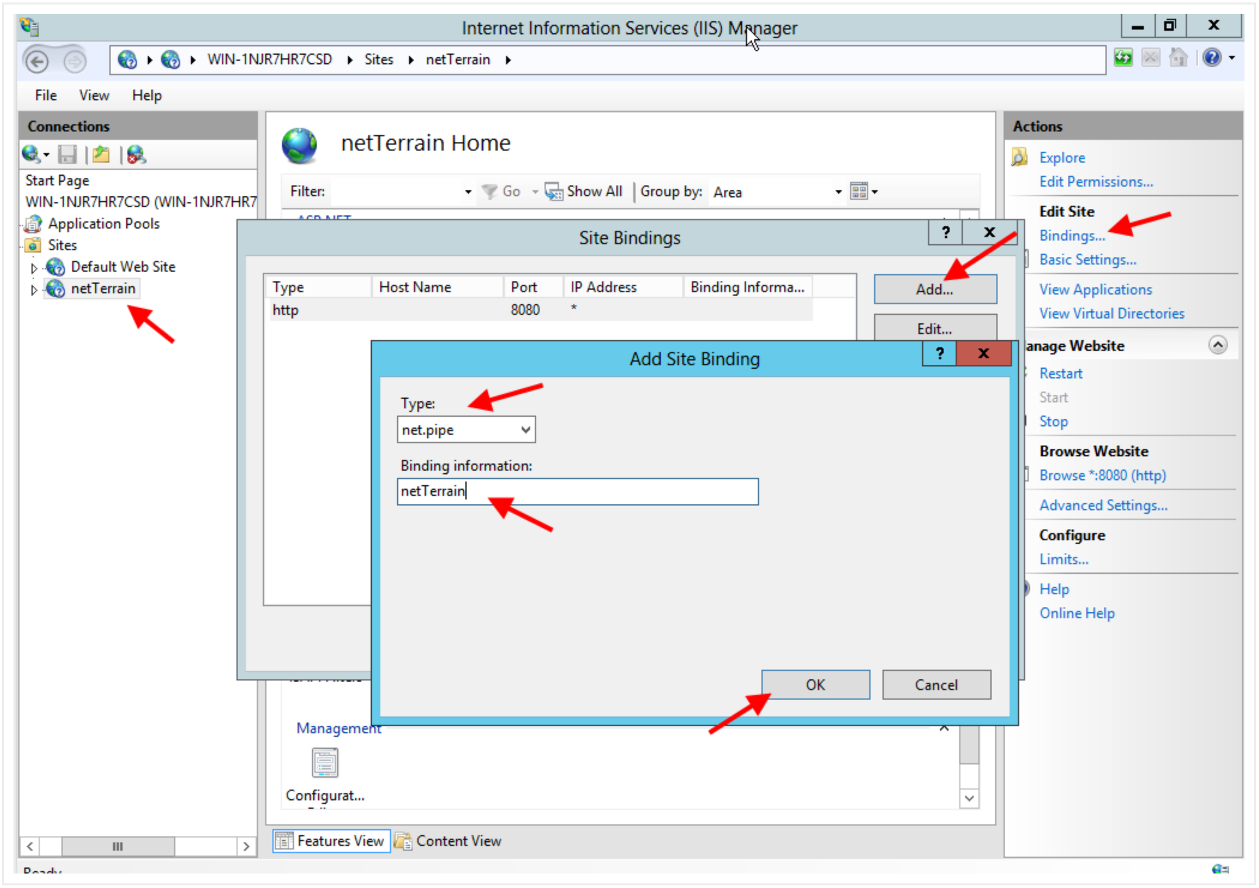
An extension module uses the so-called Named Pipes transport protocol to interact with your netTerrain server instance.

2.5.1 Setting up Named Pipes support

To set up the communication between your custom module and netTerrain we will use a transport protocol referred to as “Named Pipes” (NP).

The NP protocol relies on local transport only, thus the client application must be placed on the same machine where the netTerrain server application lives. This is not a limitation for our custom modules deployment, quite the opposite: custom modules are integrated and interact directly with the netTerrain server. Also, by leveraging local transport on the same server, custom modules do not require authentication, let alone SSL or other security features.

To turn on NP support you need to include net.pipe binding to the netTerrain site in IIS, as shown in the screenshot below. Use “netTerrain” as the value in the Binding information field.



Type	Host Name	Port	IP Address	Binding Informa...
http		8080	*	

Type: net.pipe

Binding information: netTerrain

If the net.pipe binding was already added before, make sure the binding information is correct.

As a second step, add the NP support in your web.config file (under the netTerrain/vis folder in your netTerrain app server). If you applied the contents to work with the Web API specified in the previous paragraphs (2.1 to 2.4) no further action is needed.

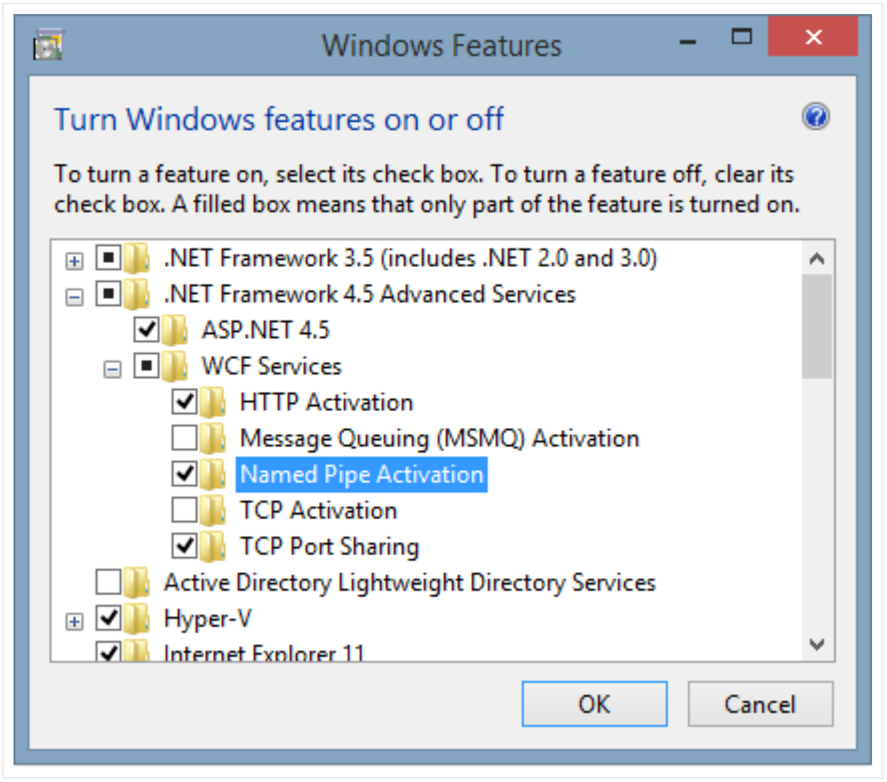
All settings related to NP are shown below. Make sure the "WebApiNetPipeAddress" key in the "appSettings" section is present.

```
<configuration>
    ...
    <appSettings>
        ...
        <add key="WebApiNetPipeAddress" value="netTerrain" />
        ...
    </appSettings>
      ...
    <system.serviceModel>
        <services>
            <service>
            ...

            <!-- Internal endpoints for Web API extension modules
access. -->

            <endpoint address="" binding="netNamedPipeBinding"
               bindingConfiguration="pipeBinding"
               contract="NetTerrain.WebApi.INetTerrainWebApi" />
            <endpoint address="pmex" binding="mexNamedPipeBinding"
               contract="IMetadataExchange"/>
            ...
            </service>
        </services>
        <bindings>
            ...
            <netNamedPipeBinding>
                <binding name="pipeBinding"
hostNameComparisonMode="Exact">
                <security mode="None" />
                </binding>
            </netNamedPipeBinding>
            ...
        </bindings>
    </system.serviceModel>
    ...
</configuration>
```

Finally, make sure the “Named Pipe Activation” windows feature is turned on:


Windows Features

Turn Windows features on or off

To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.

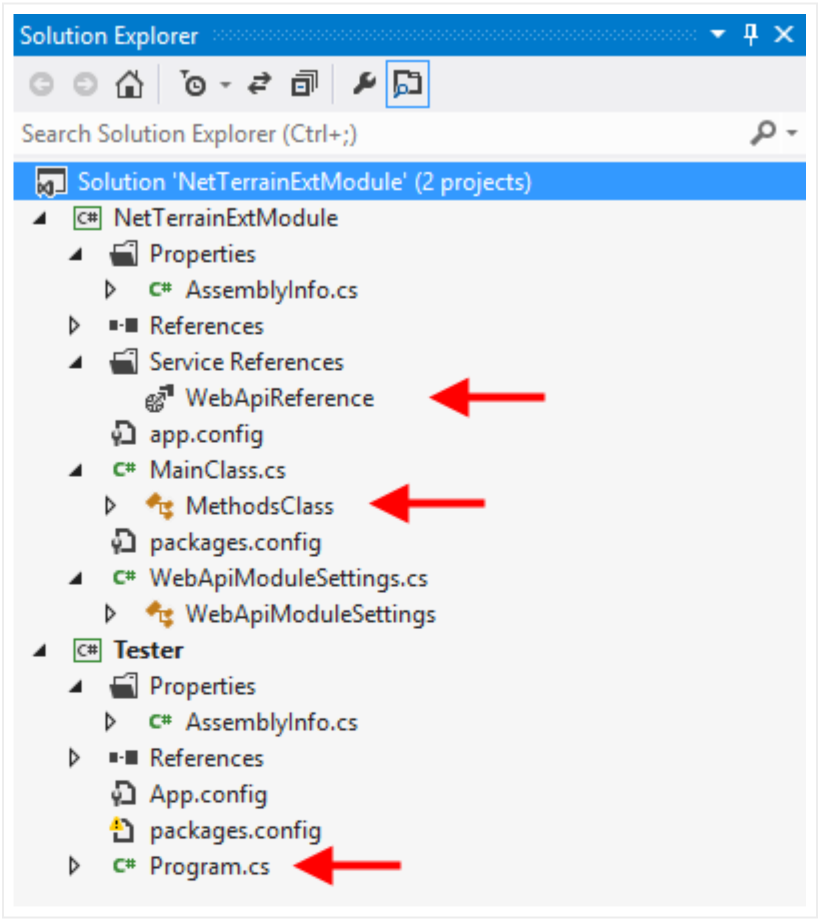
- ■ .NET Framework 3.5 (includes .NET 2.0 and 3.0)
- ■ .NET Framework 4.5 Advanced Services
  - ✓ ASP.NET 4.5
  - ■ WCF Services
    - ✓ HTTP Activation
    - ☐ Message Queuing (MSMQ) Activation
    - ✓ Named Pipe Activation
    - ☐ TCP Activation
    - ✓ TCP Port Sharing
- ☐ Active Directory Lightweight Directory Services
- ✓ Hyper-V
- ✓ Internet Explorer 11

2.5.2 Creating a sample extension module

An extension module can be written in any programming language that supports WCF and NP. As is usual in life, certain restrictions apply, so the easiest way is to write your extension in C#. It is also the language of choice in our guide. The netTerrain installer provides a sample extension module called NetTerrainExtModule, which we will refer to in the upcoming exercise. You can find it under the API folder of your netTerrain server installer, typically under C:\ProgramData\Graphical Networks\netTerrain\vis\WebApi\modules. To test it you must be able to access the netTerrain instance with a Web API call already.

2.5.2.1 Opening the solution

Unzip the contents of the Extension Module Toolkit.zip file and open the NetTerrainExtModule.sln solution in Microsoft Visual Studio 2012 (or better). The structure of the project is shown below and the more important files for our exercise are marked with red arrows.


Solution Explorer

Search Solution Explorer (Ctrl+;)

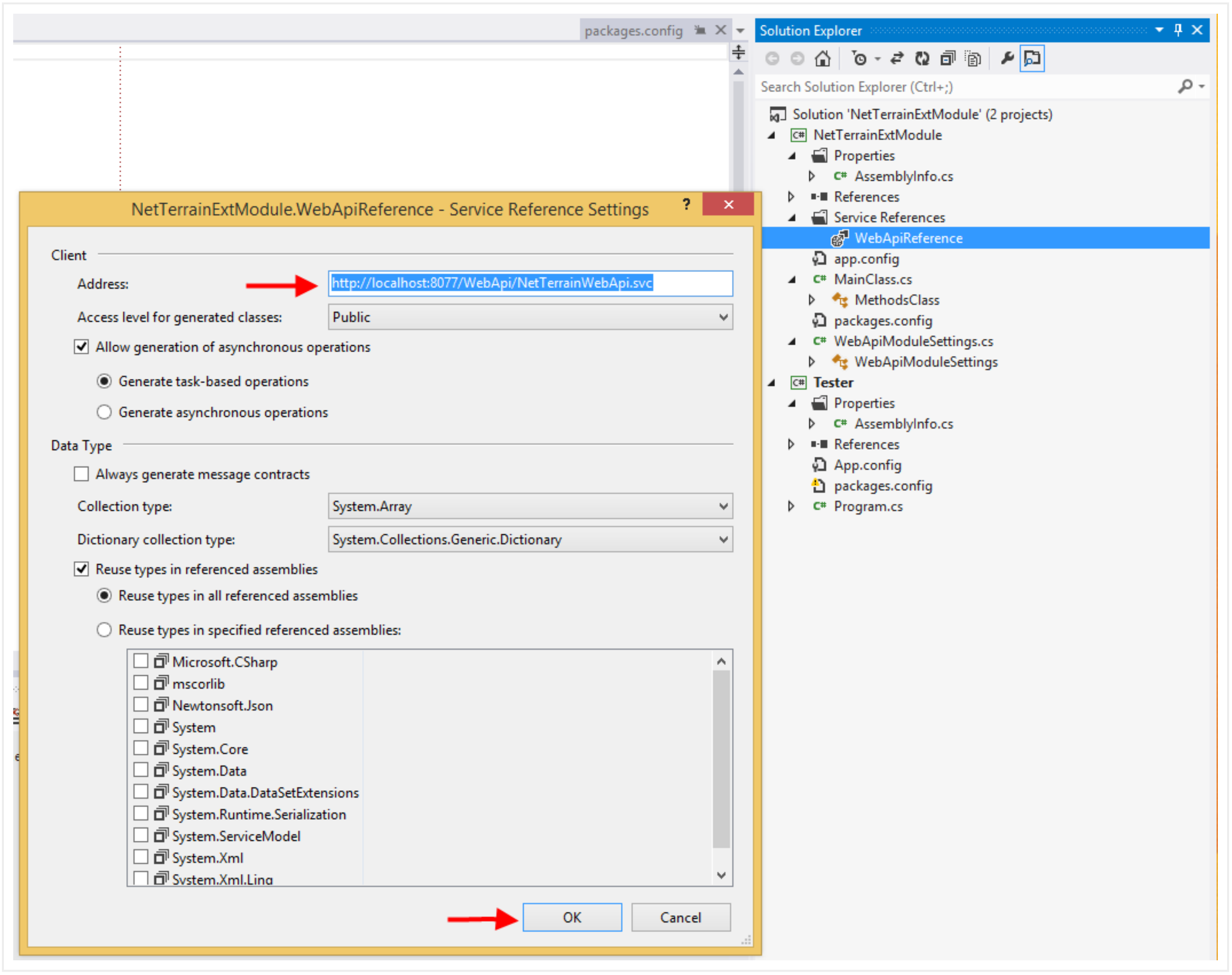
```
Solution 'NetTerrainExtModule' (2 projects)
◢ NetTerrainExtModule
   ◢ Properties
      ▷ AssemblyInfo.cs
   ▷ References
   ◢ Service References
        WebApiReference
     app.config
   ◢ MainClass.cs
      ▷ MethodsClass
     packages.config
   ◢ WebApiModuleSettings.cs
      ▷ WebApiModuleSettings
◢ Tester
   ◢ Properties
      ▷ AssemblyInfo.cs
   ▷ References
     App.config
     packages.config
   ▷ Program.cs
```

- WebApiReference: is a service reference to the netTerrain Web API. You will need to configure it to refer to a real service address you have access to.
- MethodsClass: this is the class where your extension methods will be placed.
- Program.cs: this file contains your extension module test code.

2.5.2.2 Configuring the Web API service reference

The service reference is only required during the development phase of your extension module for testing purposes. It also allows Visual Studio's IntelliSense feature to expose available methods and their signature. The service reference is not used by the extension module after its integration into netTerrain. You don't need to be concerned about specific endpoints used in the service reference since the extension module will not depend on them once placed in the actual folder of the deployed (or production) instance.

Set up the service reference by right-clicking on the WebApiReference entry in the solution explorer and select "Configure Service Reference":


NetTerrainExtModule.WebApiReference - Service Reference Settings

Client

Address: http://localhost:8077/WebApi/NetTerrainWebApi.svc

Access level for generated classes: Public

[✓] Allow generation of asynchronous operations

(●) Generate task-based operations

( ) Generate asynchronous operations

Data Type

[ ] Always generate message contracts

Collection type: System.Array

Dictionary collection type: System.Collections.Generic.Dictionary

[✓] Reuse types in referenced assemblies

(●) Reuse types in all referenced assemblies

( ) Reuse types in specified referenced assemblies:

[ ] Microsoft.CSharp
[ ] mscorlib
[ ] Newtonsoft.Json
[ ] System
[ ] System.Core
[ ] System.Data
[ ] System.Data.DataSetExtensions
[ ] System.Runtime.Serialization
[ ] System.ServiceModel
[ ] System.Xml
[ ] System.Xml.Linq

Provide a working Web API address for the existing netTerrain instance and click ‘Ok’. If the address is correct Visual Studio will update the reference accordingly. At this point you should be able to write and test your own methods.

2.5.2.3 Exploring the Methods Class

Our netTerrain extension module template encapsulates all operations related to the WCF service work and NP transport, so you only need to focus on writing your methods. To help you with this process, our MethodsClass contains a set of simple methods demonstrating different key aspects of the extension module. Feel free to choose the methods that better align with your requirements. Below is a list of methods in our sample project and an example:

- ExtensionMethodExample
- RunMethodWithParameters
- RunARealMethodAndRefresh
- Return150NoRefresh
- RunOpenGoogleUrl
- RunOpenUrlInNetTerrain
- RunTryOpenEmptyUrl
- ThrowErrorResponse
- RunMethodWithWrongInputData
- ThrowExceptionManually

```
// Extension method. Use this template to create your own methods.
// Parametes 'serviceUrl' and 'webApiUser' should always be at the
first and second places
// respectively.
public string ExtensionMethodExample(string serviceUrl, string
webApiUser)
{
    // 'CallExtensionMethod' wrapper establishes a connection with the
service and registers the
    // user. ALways use this wrapper for your methods.
    return CallExtensionMethod(serviceUrl, webApiUser, (proxy) =>
    {
    // Add your code here...
    // Call Api methods using "proxy":
    proxy.TestConnection();

    // Make a response according to method's results. Response format
is JSON string.
    // Use "true" as the second parameter to force page refresh after
method is completed.
    return CreateSuccessfulResponse("Sucessful response text.", true);
    });
}
```

Notice how this module handles all housekeeping background operations including error handling for you.

2.5.2.4 Method responses

Upon execution of an extension module, netTerrain waits for a result message. Depending on the type of result, netTerrain will trigger a specific action. These actions include:

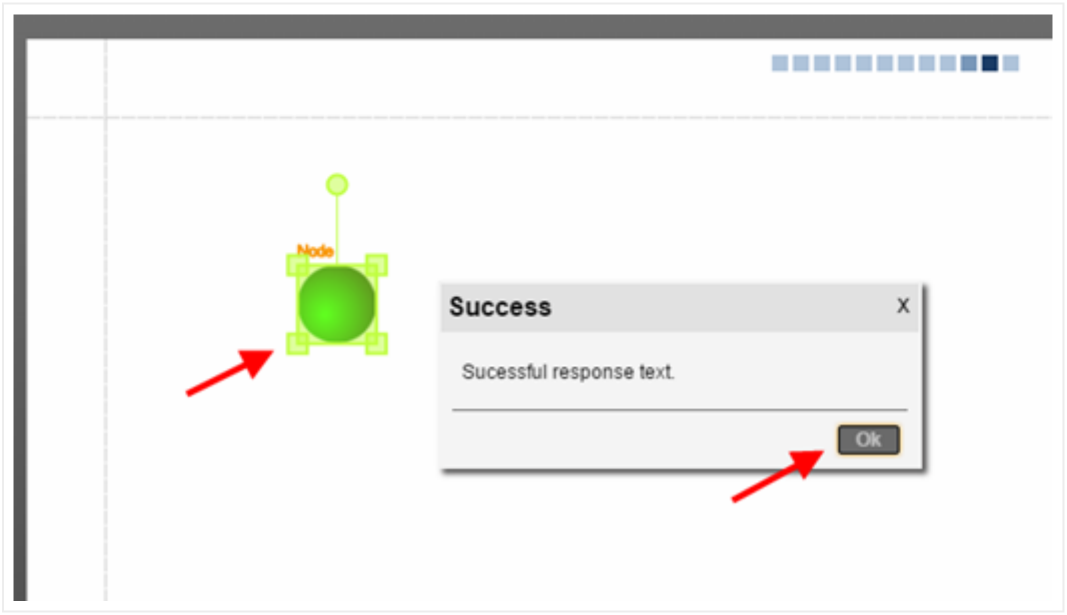
- Informing the user of a successful operation
- Reporting to the user that something went wrong
- Opening a URL
- Refreshing the netTerrain browser page

Successful response

The successful response object and syntax are as follows:

```
string CreateSuccessfulResponse(string message, bool isRefresh);
...
return CreateSuccessfulResponse("Successful response text.", true);
```

When this response is received by netTerrain it shows the user a message determined by the "message" parameter. After the user closes the message dialog, it refreshes the diagram.



After clicking on 'Ok' the page is refreshed. To prevent netTerrain from refreshing itself use the 'false' flag in the second parameter of the signature.

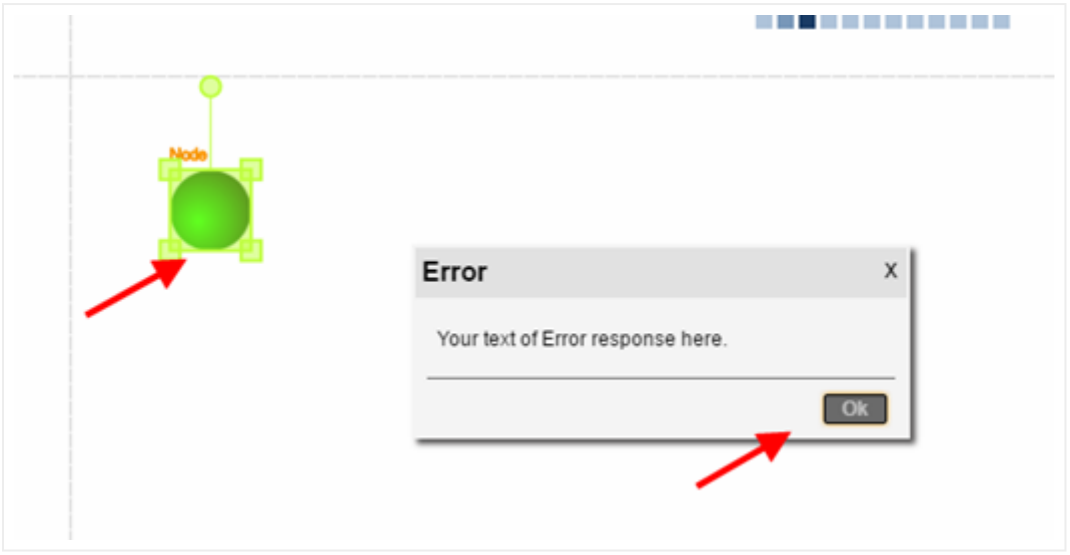
Error Response

The error response object is defined as follows:

```
string CreateErrorResponse(string error);
```

When this response is received by the netTerrain application, it shows the user an error message determined by the ‘error’ parameter. Upon closing the response dialog the page is not refreshed.

```
public string ThrowErrorResponse(string serviceUrl, string webApiUser)
{
   return CallExtensionMethod(serviceUrl, webApiUser, (proxy) =>
   {
     return CreateErrorResponse("Your text of Error response here.");
   });
}
```

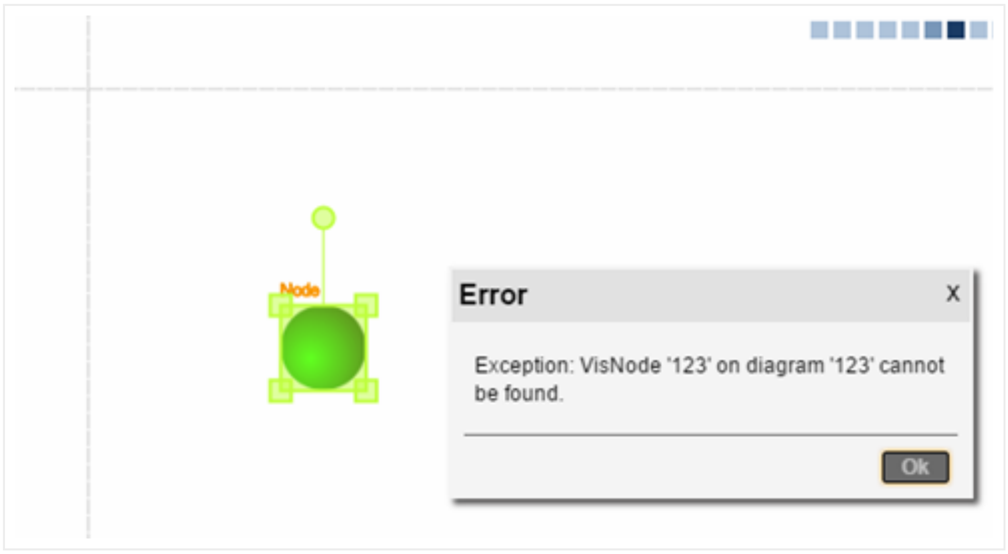


Your text of Error response here.

A similar response dialog is displayed automatically when an unexpected error occurs during the processing of an API method. In those cases, the contents of the error thrown by the netTerrain engine are displayed in the body of the error message. Below is an example of a function that forces an application level error to be thrown by netTerrain upon execution of the module.

```
public string RunMethodWithWrongInputData(string serviceUrl, string
webApiUser)
{
   return CallExtensionMethod(serviceUrl, webApiUser, (proxy) =>
   {
     proxy.NodeSetWidth(123, 123, 123);
```

```
        return CreateSuccessfulResponse("This response will never
occur.", true);
    });
}
```



Exception: VisNode '123' on diagram '123' cannot be found.

Redirection Response

The redirection response object is defined as follows:

```
string CreateSuccessfulOpenUrlResponse(string message, string url,
bool isInNewTab, bool isAddBaseUrl);
```

This is a special successful response that opens a page using the specified URL. This response does not refresh the netTerrain page.

```
public string RunOpenGoogleUrl(string serviceUrl, string webApiUser)
{
  return CallExtensionMethod(serviceUrl, webApiUser, (proxy) =>
      {
      return CreateSuccessfulOpenUrlResponse(
        "Press 'Ok' to open url...",
        @"http://google.com",
      true,
      false);
  });
}
```

Upon execution of this method, netTerrain will show a dialog prompting the user to press ‘Ok’ to open a page with the specified URL. When the third parameter is set to true, the corresponding web page will be opened in a new tab. The last parameter is set to true when an inner URL (netTerrain page on that same application) needs to be opened, such as the example below.

```
public string RunOpenUrlInNetTerrain(string serviceUrl, string
webApiUser)
{
    return CallExtensionMethod(serviceUrl, webApiUser, (proxy) =>
    {
      return CreateSuccessfulOpenUrlResponse(
      "Press 'Ok' to open url...",
      @"Catalog/ObjectOverrides",
    false,
    true);
  });
}
```

To open this page successfully, the URL is appended to the base URL.

2.5.2.5 Running Web API methods in the extension module

The extension module encapsulates all background work and exposes all API methods via a “proxy” object as depicted below:

```
public string RunARealMethodAndRefresh(string serviceUrl, string
webApiUser)
{
    return CallExtensionMethod(serviceUrl, webApiUser, (proxy) =>
    {
        proxy.NodeInsert("Scylla", 24000000000001, 26000000000000);
        proxy.NodeInsert("SomePoorGreekGalley", 24000000000001,
26000000000000);
        proxy.NodeInsert("Charybdis", 24000000000001, 26000000000000);
        return CreateSuccessfulResponse("Three nodes have been
added.", true);
    });
}
```

Using the proxy object, all methods in the API reference are available. This allows users to extend netTerrain functionality by manipulating objects in netTerrain and extend its functionality. Not too shabby.

2.5.2.6 Using parameters

Each custom module method includes two mandatory paremeters already seen before (serviceUrl and webApiUser). In addition to these two parameters, several optional parameters can be added to your method signature so that netTerrain can work with custom data. Below is an example showing a series of parameters in the method call:

```
public string RunMethodWithParameters(string serviceUrl, string
webApiUser,
  long id, long diagramId, long userId, string selection, string x,
string y,
  string containerId, string stringParam, bool boolParam, int
intParam,
  string sqlExpressionParam, float floatParam, double doubleParam,
  float floatSqlExpParam)
{
      return CallExtensionMethod(serviceUrl, webApiUser, (proxy) =>
      {
          return CreateSuccessfulResponse("Parameters: " +
          "Id(long): " + id + "\n" +
          "DiagramId(long): " + diagramId + "\n" +
          "UserId(long): " + userId + "\n" +
          "Selection(string): " + selection + "\n" +
          "X: " + x + "\n" +
          "Y: " + y + "\n" +
          "ContainerId: " + containerId + "\n" +
          "String param: " + stringParam + "\n" +
          "Bool param: " + boolParam + "\n" +
          "Int param: " + intParam + "\n" +
          "SQL Expression param: " + sqlExpressionParam + "\n" +
          "Float param: " + floatParam + "\n" +
          "Double param: " + doubleParam + "\n" +
          "Float SQL Exp param: " + floatSqlExpParam + "."
          , false);
      });
}
```

Later in this section we explore how to use all these parameters and pass them in netTerrain.

2.5.2.7 Testing extension module methods

A testing console application is included in our sample solution containing a “Program.cs” file. It provides a simple testing example, as shown below:

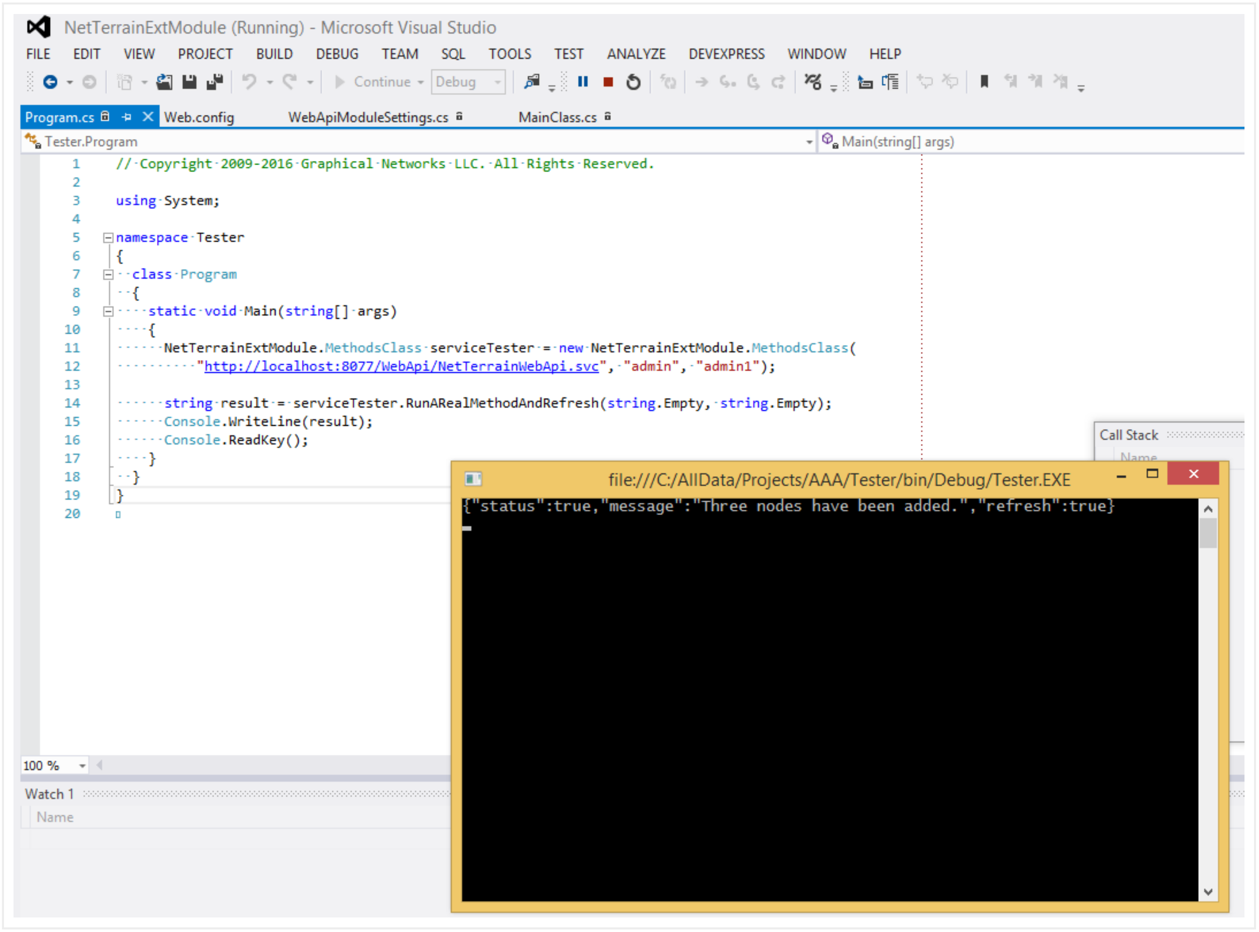
```
static void Main(string[] args)
{
    NetTerrainExtModule.MethodsClass serviceTester =
    new NetTerrainExtModule.MethodsClass(
    "http://localhost:8077/WebApi/NetTerrainWebApi.svc",
    "serviceUser",
    "servicePassword");
    string result =
serviceTester.RunARealMethodAndRefresh(string.Empty, string.Empty);
    Console.WriteLine(result);
    Console.ReadKey();
}
```

Note that the first and second parameters of the API method consist of empty strings. Leave them empty whenever you are using the module in test mode. netTerrain will automatically use them after the integration of the extension module into its corresponding instance.

Also note that for testing purposes you need proper netTerrain Web API and service credentials. This is only the case for testing, once integrated into netTerrain, the module utilizes the credentials of the user invoking the module.

Be careful when testing an extension module as you could still be affecting an actual netTerrain instance!

In the image below, we show the results of our test case above, after being executed:


NetTerrainExtModule (Running) - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ANALYZE DEVEXPRESS WINDOW HELP
Continue
Debug
Program.cs
Web.config
WebApiModuleSettings.cs
MainClass.cs
Tester.Program
Main(string[] args)
Call Stack
Name
100 %
Watch 1
Name
file:///C:/AllData/Projects/AAA/Tester/bin/Debug/Tester.EXE

```
// Copyright 2009-2016 Graphical Networks LLC. All Rights Reserved.

using System;

namespace Tester
{
  class Program
  {
    static void Main(string[] args)
    {
      NetTerrainExtModule.MethodsClass serviceTester = new NetTerrainExtModule.MethodsClass(
          "http://localhost:8077/WebApi/NetTerrainWebApi.svc", "admin", "admin1");

      string result = serviceTester.RunARealMethodAndRefresh(string.Empty, string.Empty);
      Console.WriteLine(result);
      Console.ReadKey();
    }
  }
}
```

```
{"status":true,"message":"Three nodes have been added.","refresh":true}
```

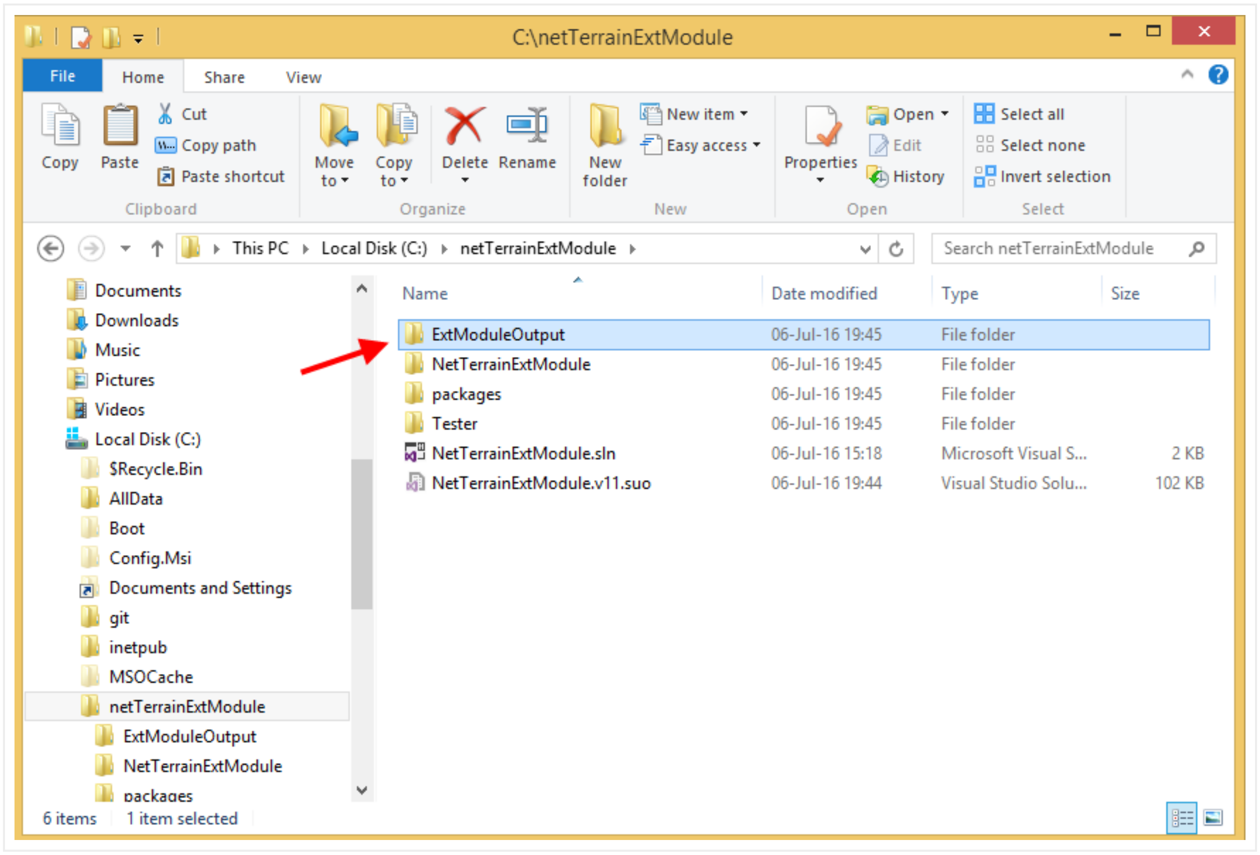
The extension module methods return responses in JSON format, such as the following output:

```
{
     "status":true,
     "message":"Three nodes have been added.",
     "refresh":true
}
```

You can analyze this output to check if your method works as expected.

2.5.2.8 Integrating the extension module into netTerrain

After successful testing of your method, we recommend rebuild your project in “Release” mode and fetching the resulting DLL file - in this case the “NetTerrainExtModule.dll” file - found under the “ExtModuleOutput” folder in your solution.



C:\netTerrainExtModule

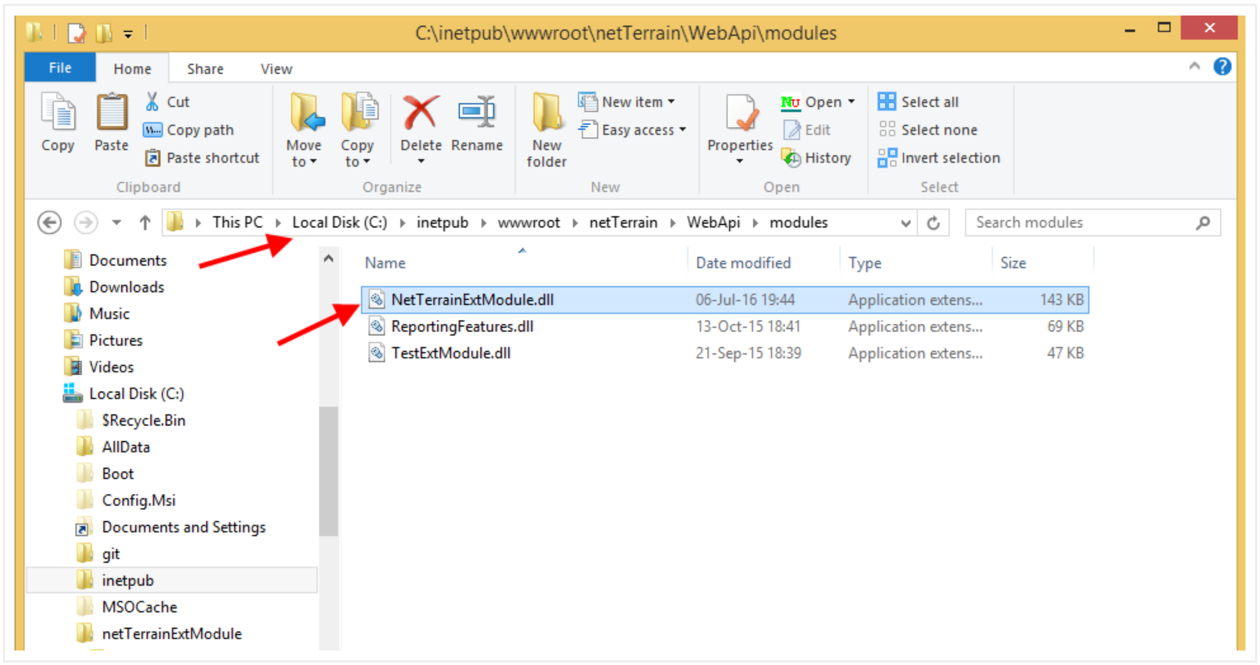
This PC ▸ Local Disk (C:) ▸ netTerrainExtModule ▸

Search netTerrainExtModule

Name	Date modified	Type	Size
ExtModuleOutput	06-Jul-16 19:45	File folder	
NetTerrainExtModule	06-Jul-16 19:45	File folder	
packages	06-Jul-16 19:45	File folder	
Tester	06-Jul-16 19:45	File folder	
NetTerrainExtModule.sln	06-Jul-16 15:18	Microsoft Visual S...	2 KB
NetTerrainExtModule.v11.suo	06-Jul-16 19:44	Visual Studio Solu...	102 KB

6 items 1 item selected

You can rename the DLL file and integrate it into your netTerrain instance. To do so, copy the file to the "WebApi/modules" folder under your netTerrain application instance path.



C:\inetpub\wwwroot\netTerrain\WebApi\modules

This PC ▸ Local Disk (C:) ▸ inetpub ▸ wwwroot ▸ netTerrain ▸ WebApi ▸ modules

Search modules

Name	Date modified	Type	Size
NetTerrainExtModule.dll	06-Jul-16 19:44	Application extens...	143 KB
ReportingFeatures.dll	13-Oct-15 18:41	Application extens...	69 KB
TestExtModule.dll	21-Sep-15 18:39	Application extens...	47 KB

Now your module is integrated! You can use this within any netTerrain instance, the only requirement being that the Web API version you use matches the version of the targeted netTerrain instance.

2.5.3 Using the extension module from the netTerrain GUI

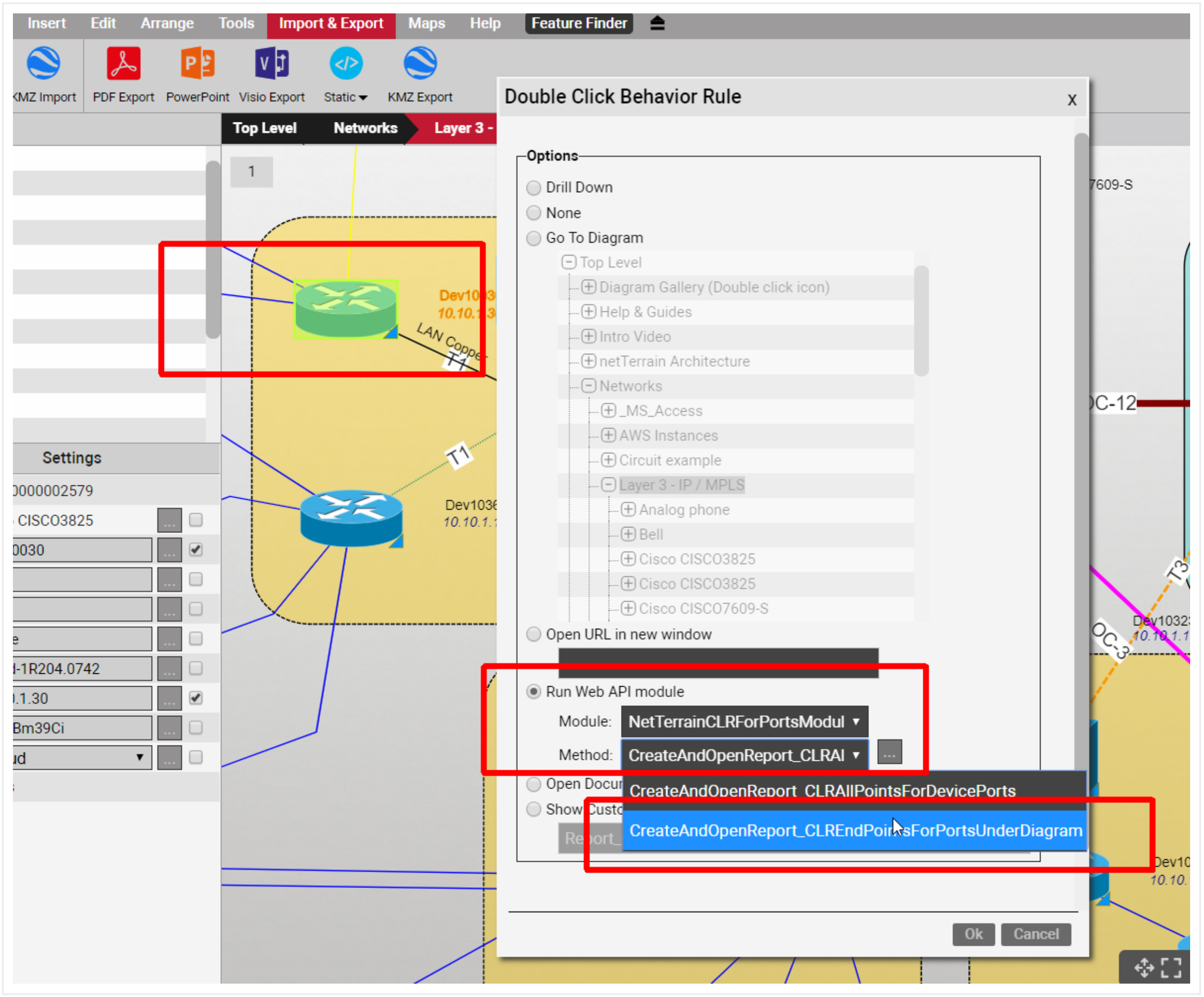
After all this work creating our extension module, we want to make sure our end users can take advantage of it.

There are two ways to utilize extension modules within the netTerrain GUI:

- Through node double-click behaviors
- Inside the custom Actions menu in the diagram or object context menu

2.5.3.1 Setting up a double-click custom action

Log into netTerrain and select a node you want to use to run the extension module. Right click and select “Double Click Behavior”. Click on “Run Web API module” and select your module and a method you want to run:



Double Click Behavior Rule

Options

- Drill Down
- None
- Go To Diagram
  - Top Level
    - Diagram Gallery (Double click icon)
    - Help & Guides
    - Intro Video
    - netTerrain Architecture
    - Networks
      - _MS_Access
      - AWS Instances
      - Circuit example
      - Layer 3 - IP / MPLS
        - Analog phone
        - Bell
        - Cisco CISCO3825
        - Cisco CISCO3825
        - Cisco CISCO7609-S
- Open URL in new window
- Run Web API module

Module: NetTerrainCLRForPortsModul

Method: CreateAndOpenReport_CLRAl

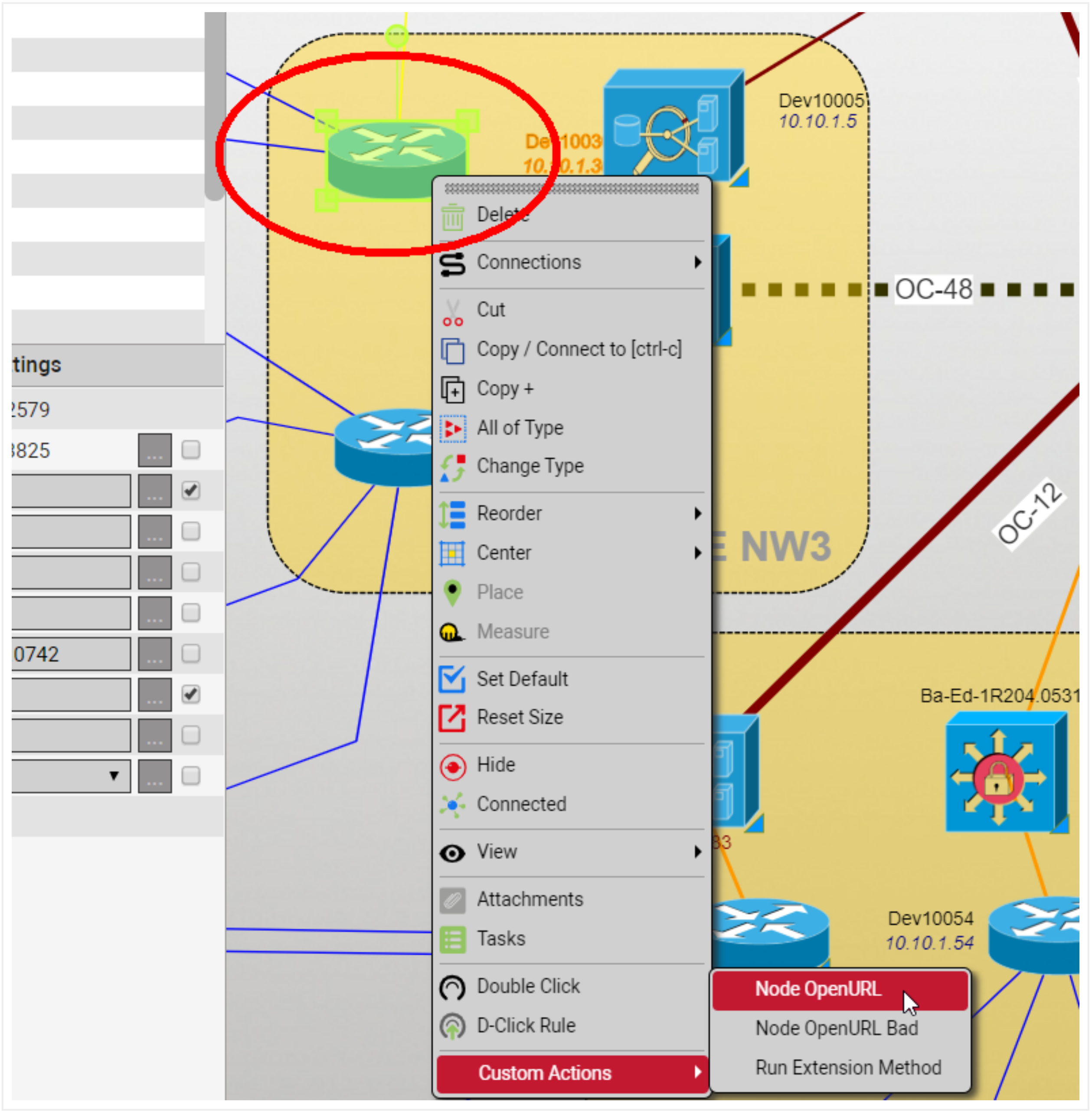
- CreateAndOpenReport_CLRAllPointsForDevicePorts
- CreateAndOpenReport_CLREndPointsForPortsUnderDiagram

Ok Cancel

Click “Ok” to close the dialog and double-click the node to run the selected method. netTerrain will show you a message with the result after processing the method.

2.5.3.2 Setting up a custom action

Custom Actions are also accessible from a diagram or object right-click context menu.



All custom actions are defined in an xml file located in C:\ProgramData\Graphical Networks\netTerrain\vis\CustomActions\CustomUserActions.xml

Below is an example of a custom action xml definition:

```
<CustomUserAction>
    <Name>Display Name Goes Here</Name>
    <Action>runextmethod</Action>
    <Filter>nodes,links,diagrams</Filter>
    <Parameters>
```

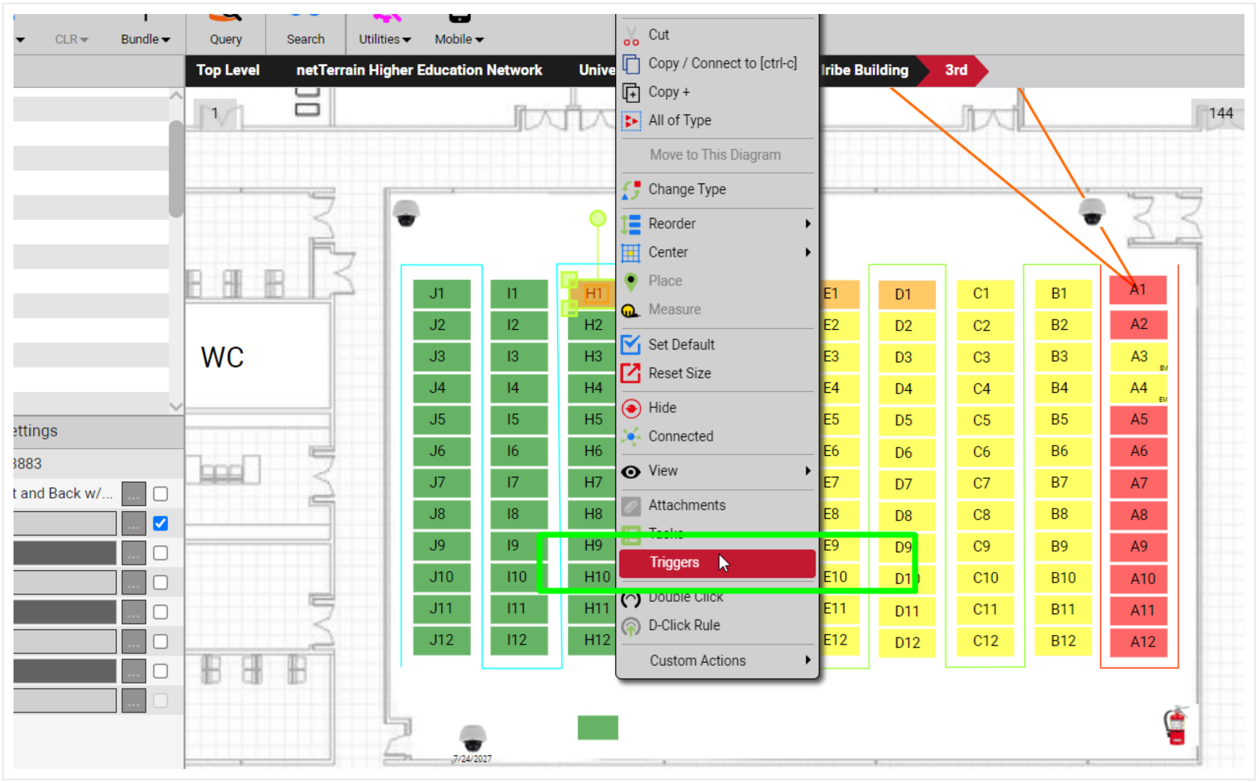
```
        <Module>NetTerrainExtModule.dll</Module>
        <Method>RunMethodWithParameters</Method>
        <MethodParameters>
            <Id>[Id]</Id>
            <DiagramId>[DiagramId]</DiagramId>
            <UserId>[UserId]</UserId>
            <Selection>[Selection]</Selection>
            <X>[X]</X>
            <Y>[Y]</Y>
            <ContainerId>[ContainerId]</ContainerId>
            <StringParam>My String Parameter</StringParam>
            <BoolParam>true</BoolParam>
            <IntParam>7298</IntParam>
            <SQLExpParam>$atest</SQLExpParam>
            <FloatParam>-677</FloatParam>
            <DoubleParam>434754467298</DoubleParam>
            <DoubleSQLExpParam>$atest</DoubleSQLExpParam>
        </MethodParameters>
    </Parameters>
</CustomUserAction>
```

In the next section we will review the parameters that can be used in the context of a custom module.

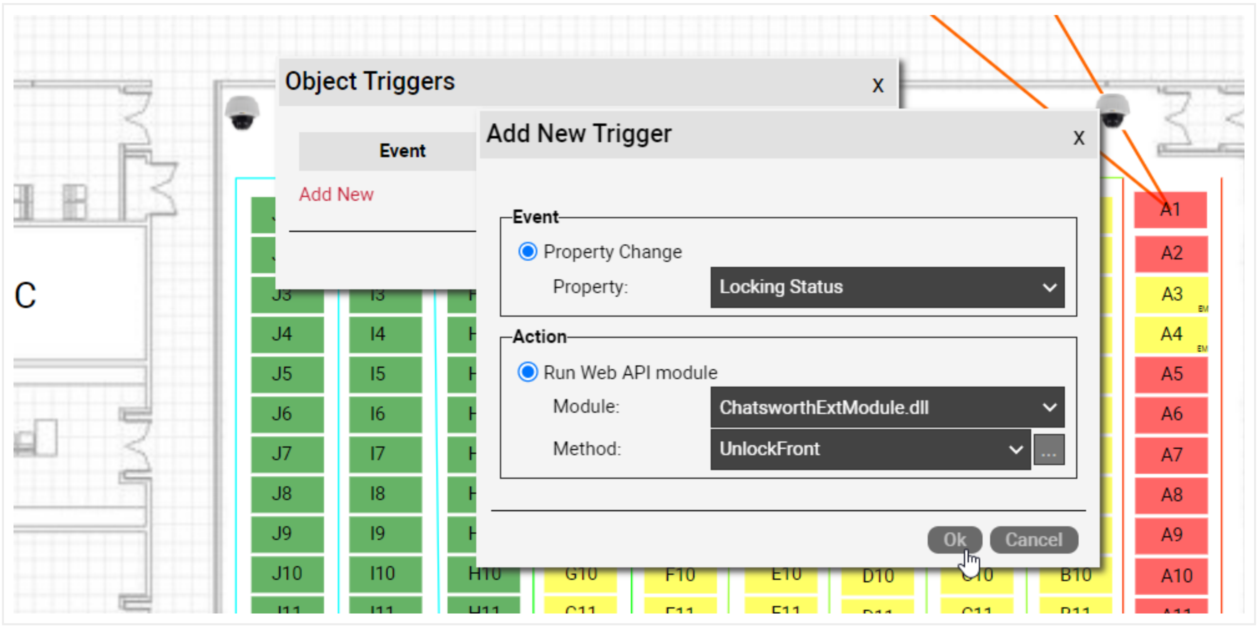
2.5.3.3 Setting up a trigger

You can set up triggers in netTerrain for any node instance. A trigger can execute a method inside a custom dll, which will be triggered upon the event being fired. Currently we only support the property change event. This event is automatically triggered by the netTerrain server when the property on the object instance for which the trigger was defined experiences a change in value.

To use the property change event first right click on the desired node that is supposed to trigger the event and click on 'Triggers':



In the 'object triggers' dialog box click on Add New to define your new property change trigger event:


Object Triggers

Event

Add New

Add New Trigger

Event

(●) Property Change

Property: Locking Status

Action

(●) Run Web API module

Module: ChatsworthExtModule.dll

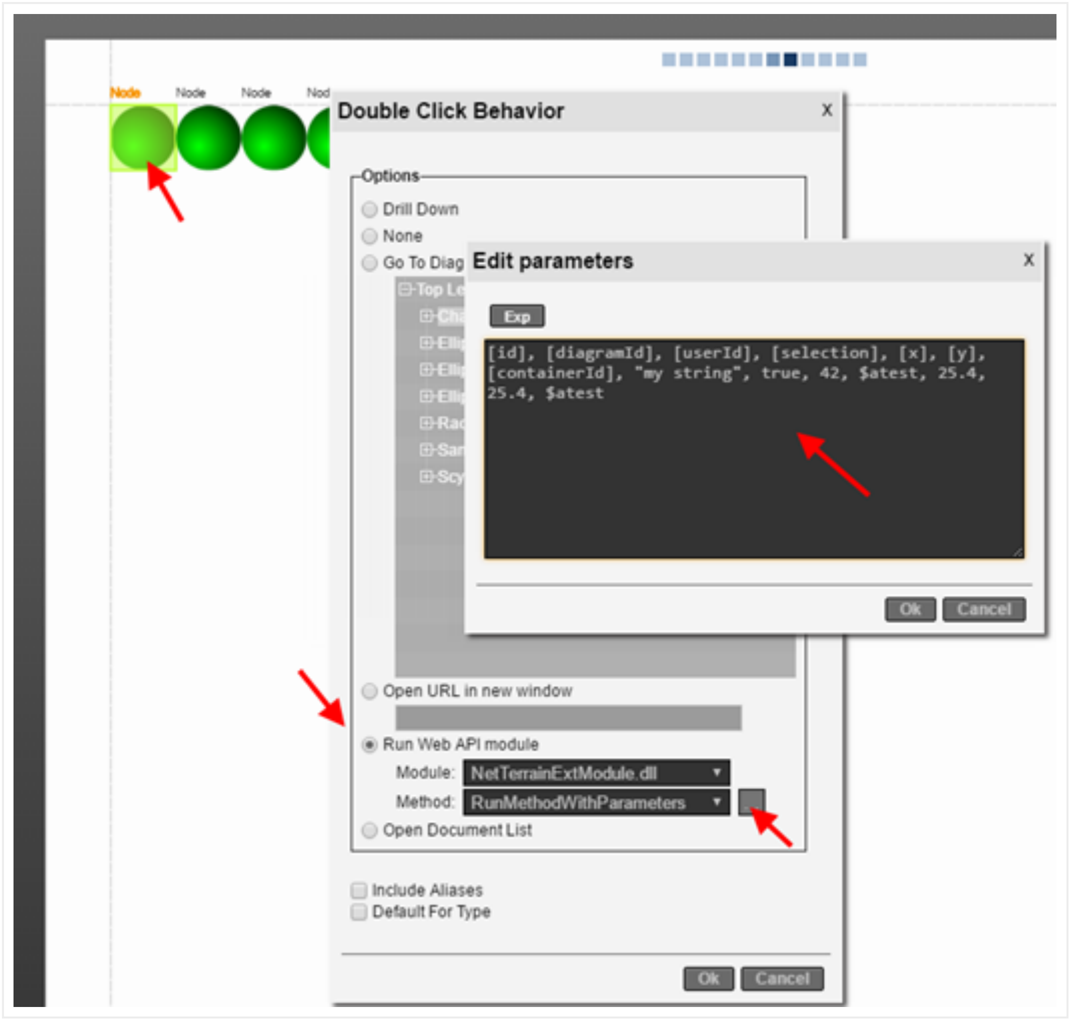
Method: UnlockFront

Ok Cancel

First, choose the object property that will trigger the event and then for the 'Action', pick the custom module and method that should be executed upon the property change.

2.5.4 Using parameters

netTerrain can pass parameters of different types to a custom module method. The previous showed an example of how to use parameters in the custom action xml definition. The image below shows an example of how to edit paremeters when invoking a custom module through a double-click action override:

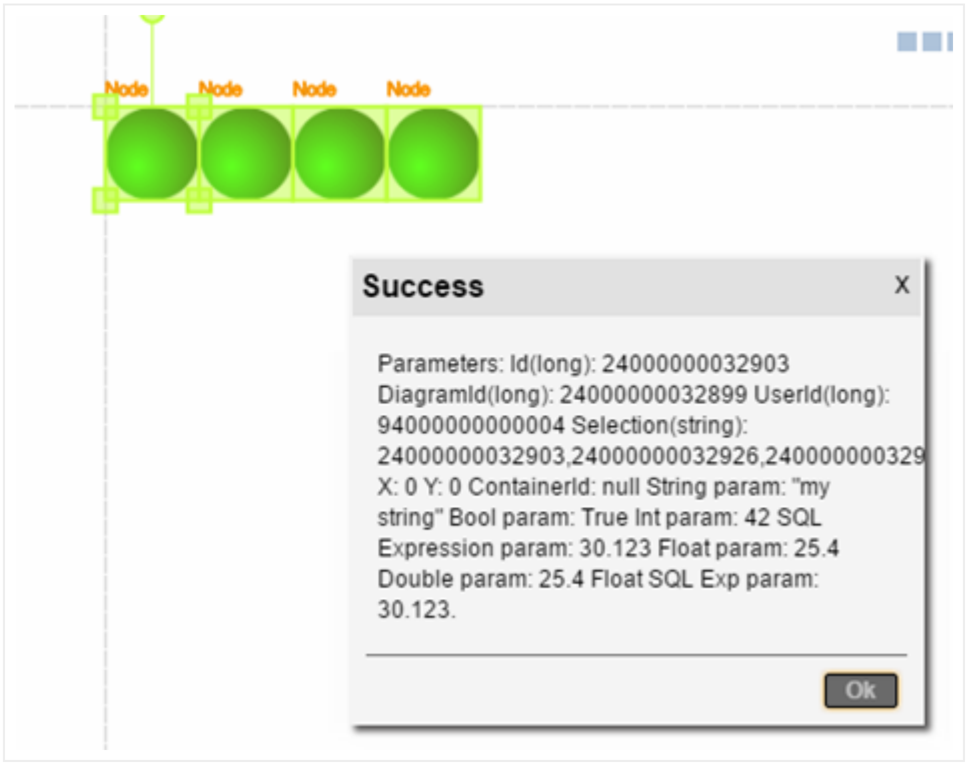


Options
- ( ) Drill Down
- ( ) None
- ( ) Go To Diag
- ( ) Open URL in new window
- (●) Run Web API module
  - Module: NetTerrainExtModule.dll
  - Method: RunMethodWithParameters
- ( ) Open Document List

- [ ] Include Aliases
- [ ] Default For Type

```
[id], [diagramId], [userId], [selection], [x], [y],
[containerId], "my string", true, 42, $atest, 25.4,
25.4, $atest
```

The result of executing the above method with the provided parameters is shown below:



Parameters: Id(long): 24000000032903 DiagramId(long): 24000000032899 UserId(long): 94000000000004 Selection(string): 24000000032903,24000000032926,240000000329 X: 0 Y: 0 ContainerId: null String param: "my string" Bool param: True Int param: 42 SQL Expression param: 30.123 Float param: 25.4 Double param: 25.4 Float SQL Exp param: 30.123.

All the accepted parameter types are defined in the extension method.

```
public string RunMethodWithParameters(
 string serviceUrl,
 string webApiUser,
 long id,
 long diagramId,
 long userId,
 string selection,
 string x,
 string y,
 string containerId,
 string stringParam,
 bool boolParam,
 int intParam,
 string sqlExpressionParam,
 float floatParam,
 double doubleParam,
 float floatSqlExpParam
);
```

The following table shows a set of sample parameter data, its associated type, computed value and defintion:

Parameter	Type	Value	Definition
id	long	24000000032903	Id of the current node (for example, the node a user double-clicked on to run the custom module method)
diagramId	long	24000000032899	Id of the current diagram
userId	long	94000000000004	Current user Id
selection	string	“24000000032903, 24000000032926, 24000000032927, 24000000032928”	A set of Ids of currently selected nodes. The set Is represented as a string.
x	string	0	X coordinate of the current node
y	string	0	Y coordinate of the current node
containerId	string	“null”	If the corrent node is mounted in some rack container this parameter contains the id of this container. Otherwise it is null.
"my string"	string	“my string”	Constant string
true	bool	true	Constant bool value
42	int	42	Constant int value
$atest	string	“30.123”	netTerrain SQL expression (please check the database description and scripting guide for more details about expressions). In this case the method returns the value 30.123\. The result is represented as a string.
25.4	float	25.4	Constant float value
25.4	double	25.4	Constant double value
$atest	float	30.123	Same SQL expression as above, this time returning the result as a float value.

3 Main API Classes

3.1 NetTerrain.WebApi Namespace

The NetTerrain.WebApi namespace contains several enumerations and classes and also provides the INetTerrainWebApi interface containing netTerrain's Web API methods.

3.1.1 Classes

	Class	Description
	FaultInfo	This class contains fault information including a description of what caused the fault during the Web API method execution. If the fault was caused by an exception in netTerrain, FaultInfo provides exception details.
	FileUploadAttributes	This class combines parameters for uploading a file to the server.
	Group	This class contains parameters of a user group.
	LinkCategory	This class contains parameters of a link category.
	LinkOverride	This class contains parameters of a link property override.
	LinkType	This class represents a link type object.
	LinkTypeProperty	This class contains parameters of a link type property.
	NewNodeTypeInfo	This class combines parameters of a new node type in conjunction with the CatalogAddNodeType(NewNodeTypeInfo) method - an older version of the node type creation method for versions compatibility. It is adviced to use the newer method [CatalogNodeTypeAdd(String, NodeTypeGroups, String, Boolean)](#459-catalognodetypeadd-method).
	NodeCategory	This class contains parameters of a node category.
	NodeOverride	This class contains parameters of a node property override.

	Class	Description
	NodeType	This class represents a node type object.
	NodeTypeProperty	This class contains parameters of a node type property.
	User	This class contains parameters of a user.
	Vendor	This class contains parameters of a vendor. Applicable for node types as Racks, Devices or Cards.

3.1.2 Interfaces

	Interface	Description
	INetTerrainWebApi	The INetTerrainWebApi interface provides all methods exposed in the netTerrain Web API. Ultimately, these are the calls that API users will be utilizing most of the time to build external applications that interact with netTerrain.

3.1.3 Enumerations

	Enumeration	Description
	ArrowStyles	This enumeration type sets link arrow styles.
	DoubleClickBehaviors	This enumeration type sets the double-click behavior of a node.
	FontFamilies	This enumeration type sets available text font families.
	HierarchySearchModes	This enumeration type sets how deeply nodes should be searched in the hierarchy tree.
	InstanceEffects	This enumeration type sets effects for an overriden instance.
	LinkStyles	This enumeration type sets styles of lines used to display links.
	NodeTypeGroups	

	Enumeration	Description
		This enumeration type describes groups of node types used in netTerrain.
	OverrideRules	This enumeration type sets override rules for node and link properties.
	Roles	This enumeration type sets group roles.
	TextAligns	This enumeration type sets horizontal alignment for multiline text within a field.
	TextJustification	This enumeration type sets the text field justification point.
	UpwardsPropagations	This enumeration type sets the upwards propagation effect applied to an overridden instance.

3.2 ArrowStyles Enumeration

This enumeration type sets link arrow styles.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

3.2.1 Syntax

C#

```
public enum ArrowStyles
```

3.2.2 Members

Member name	Value	Description
Plain	0	Plain.
Reversed	1	Reversed.
Solid	2	Solid.
SolidReversed	3	Solid Reversed.
Open	4	Open.

Member name	Value	Description
OpenReversed	5	Open Reversed.
Triangle	6	Triangle.
TriangleReversed	7	Triangle Reversed.
TriangleSolid	8	Triangle Solid.
TriangleSolidReversed	9	Triangle Solid Reversed.
Diamond	10	Diamond.
DiamondSolid	11	Diamond Solid.
Square	12	Square.
SquareSolid	13	Square Solid.
Circle	14	Circle.
CircleSolid	15	Circle Solid.

3.2.3 See Also

NetTerrain.WebApi Namespace

3.3 DoubleClickBehaviors Enumeration

This enumeration type sets the double-click behavior of a node.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

3.3.1 Syntax

C#

```
public enum DoubleClickBehaviors
```

3.3.2 Members

Member name	Value	Description
DrillDown	0	Drill down.
None	1	Do nothing.
GoToDiagram	2	Go to a specific diagram.
OpenURL	3	Open a specified URL.

3.3.3 See Also

NetTerrain.WebApi Namespace

3.4 FaultInfo Class

This class contains fault information including a description of what caused the fault during the Web API method execution. If the fault was caused by an exception in netTerrain, **FaultInfo** provides exception details.

3.4.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.FaultInfo

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

3.4.2 Syntax

C#

```
public class FaultInfo
```

The **FaultInfo** type exposes the following members.

3.4.3 Constructors

	Name	Description
	FaultInfo	

3.4.4 Fields

	Name	Description
	Description	Fault description.
	Details	Full content of the exception thrown in netTerrain.
	Message	Exception message thrown in netTerrain.
	Type	Type of the exception thrown in netTerrain.

3.4.5 See Also

NetTerrain.WebApi Namespace

3.5 FileUploadAttributes Class

This class combines parameters for uploading a file to the server.

3.5.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.FileUploadAttributes

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

3.5.2 Syntax

C#

```
public class FileUploadAttributes : IDisposable
```

The **FileUploadAttributes** type exposes the following members.

3.5.3 Constructors

	Name	Description
	FileUploadAttributes	

3.5.4 Methods

	Name	Description
	Dispose	Closes file byte stream if opened.

3.5.5 Fields

	Name	Description
	FileByteStream	Byte stream for uploading the file to the server.
	FileName	Uploaded file name.
	ObjectId	Id of the related object the file is intended for (Node type, node override etc).

3.5.6 See Also

NetTerrain.WebApi Namespace

3.6 FontFamilies Enumeration

This enumeration type sets the available text font families.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

3.6.1 Syntax

C#

```
public enum FontFamilies
```

3.6.2 Members

Member name	Value	Description
Arial	0	Arial.
ArialBlack	1	Arial Black.
ComicSansMS	2	Comic Sans MS.
CourierNew	3	Courier New.
Georgia	4	Georgia.
PalatinoLinotype	5	Palatino Linotype.
Tahoma	6	Tahoma.
TimesNewRoman	7	Times New Roman.
TrebuchetMS	8	Trebuchet MS.
LucidaSansUnicode	9	Lucida Sans Unicode.
LucidaConsole	10	Lucida Console.
MSSerif	11	MS Serif.
Verdana	12	Verdana.

3.6.3 See Also

NetTerrain.WebApi Namespace

3.7 Group Class

This class contains parameters of a user group.

3.7.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.Group

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

3.7.2 Syntax

C#

```
public class Group
```

The **Group** type exposes the following members.

3.7.3 Constructors

	Name	Description
	Group	

3.7.4 Fields

	Name	Description
	Id	Group id.
	IsSystem	If set to true indicates that the group belongs to system groups list and cannot be altered or deleted.
	Name	Group name.
	Role	Group role according to permissions for group members.

3.7.5 See Also

NetTerrain.WebApi Namespace

3.8 HierarchySearchModes Enumeration

This enumeration type sets how deeply nodes should be searched in the hierarchy tree.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

3.8.1 Syntax

C#

```
public enum HierarchySearchModes
```

3.8.2 Members

Member name	Value	Description
DiagramOnly	0	Search on a specified diagram only.
N_Levels	1	Seach up to N levels down inclusively.
Full	2	search throughout all levels.

3.8.3 See Also

NetTerrain.WebApi Namespace

4 API Reference

4.1 INetTerrainWebApi Interface

The INetTerrainWebApi interface provides all methods exposed in the netTerrain Web API. Ultimately, these are the calls that API users will be utilizing most of the time to build external applications that interact with netTerrain.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.1.1 Syntax

C#

```
public interface INetTerrainWebApi
```

The **INetTerrainWebApi** type exposes the following members.

4.1.2 Methods

	Name	Description
	AdminGroupAdd	This method adds a new user group.
	AdminGroupDelete	This method deletes a user group.
	AdminGroupGet	This method gets a user group.
	AdminGroupGetByName	This method gets a user group by name.
	AdminGroupUpdate	This method updates the user group.
	AdminUserAdd	This method adds a new user.
	AdminUserDelete	This method deletes a user.
	AdminUserGet	This method gets a user.
	AdminUserGetByName	This method gets a user by name.
	AdminUserSetPassword	This method changes user password.
	AdminUserUpdate	This method updates the user.
	CatalogAddLinkType	This method adds a new link type to the catalog. This is an older version of the method. It is adviced to use the newer method [CatalogLinkTypeAdd(String, Boolean)
	CatalogAddLinkTypeProperty	This method adds a new property to an existing link type in the catalog. This is an older version of the [CatalogLinkTypePropertyAdd(Int64, String,

	Name	Description
		String, Boolean, Boolean, Boolean) method. The new method was added to follow the current method naming convention.
	CatalogAddNodeType	This method adds a new node type to the catalog. This is an older version of the method. It is adviced to use the newer method [CatalogNodeTypeAdd(String, NodeTypeGroups, String, Boolean) with better handling of the TypeGroupNumber parameter.
	CatalogAddNodeTypeProperty	This method adds a new property to an existing node type in the catalog. This is an older copy of [CatalogNodeTypePropertyAdd(Int64, String, String, Boolean, Boolean, Boolean) method. The new method was added to follow the current method naming convention.
	CatalogGetLinkTypeProperties	This method gets a list of properties for a link type.
	CatalogGetNodeTypeProperties	This method gets a list of properties for a node type.
	CatalogLinkCategoryAdd	This method adds a new link category.
	CatalogLinkCategoryDelete	This method deletes a link category from the catalog.
	CatalogLinkCategoryGet	This method returns a link category.
	CatalogLinkCategoryGetByName	This method returns a link category providing its name.
	CatalogLinkCategoryUpdate	This method updates the link category in the catalog.
	CatalogLinkCategoryUploadImage	This method uploads a new icon image for an existing link category.
	CatalogLinkOverrideAdd	This method adds a new override for a link type property.
	CatalogLinkOverrideAddListValue	This method adds a new list value for a link type property.
	CatalogLinkOverrideDelete	

	Name	Description
		This method deletes a link override (list value) from the catalog.
	CatalogLinkOverrideGet	This method returns a link override.
	CatalogLinkOverrideGetByListValue	This method returns a link override providing its list value and link property id.
	CatalogLinkOverridesGetByPropertyId	Gets a list of link overrides for a specified link type property.
	CatalogLinkOverridesGetByTypeId	Gets a list of link overrides for all properties of a specified link type.
	CatalogLinkOverrideUpdate	This method updates the link override in the catalog.
	CatalogLinkTypeAdd	This method adds a new link type to the catalog.
	CatalogLinkTypeClone	This method clones an existing link type in the catalog.
	CatalogLinkTypeDelete	This method deletes a link type from the catalog.
	CatalogLinkTypeGet	This method returns a link type.
	CatalogLinkTypeGetByName	This method returns a link type providing its name.
	CatalogLinkTypePropertyAdd	This method adds a new property to an existing link type in the catalog.
	CatalogLinkTypePropertyDelete	This method deletes a link type property from the catalog.
	CatalogLinkTypePropertyGet	This method returns a link type property.
	CatalogLinkTypePropertyGetByName	This method returns a link type property providing its name.
	CatalogLinkTypePropertyUpdate	This method updates the link type property in the catalog.
	CatalogLinkTypeUpdate	This method updates the link type in the catalog.
	CatalogNodeCategoryAdd	This method adds a new node category.

	Name	Description
	CatalogNodeCategoryDelete	This method deletes a node category from the catalog.
	CatalogNodeCategoryGet	This method returns a node category.
	CatalogNodeCategoryGetByName	This method returns a node category providing its name.
	CatalogNodeCategoryUpdate	This method updates the node category in the catalog.
	CatalogNodeCategoryUploadImage	This method uploads a new icon image for an existing node category.
	CatalogNodeOverrideAddtd>	This method adds a new override for a node type property.
	CatalogNodeOverrideAddListValue	This method adds a new list value for a node type property.
	CatalogNodeOverrideDelete	This method deletes a node override (list value) from the catalog.
	CatalogNodeOverrideGet	This method returns a node override.
	CatalogNodeOverrideGetByListValue	This method returns a node override providing its list value and node property id.
	CatalogNodeOverridesGetByPropertyId	Gets a list of node overrides for a specified node type property.
	CatalogNodeOverridesGetByTypeId	Gets a list of node overrides for all properties of a specified node type.
	CatalogNodeOverrideUpdate	This method updates the node override in the catalog.
	CatalogNodeOverrideUploadImage	This method uploads a new override image for an existing node override.
	CatalogNodeTypeAdd	This method adds a new node type to the catalog.
	CatalogNodeTypeClone	This method clones an existing node type in the catalog.
	CatalogNodeTypeDelete	This method deletes a node type from the catalog.

	Name	Description
	CatalogNodeTypeGet	This method returns a node type.
	CatalogNodeTypeGetByName	This method returns a node type providing its name.
	CatalogNodeTypePropertyAdd	This method adds a new property to an existing node type in the catalog.
	CatalogNodeTypePropertyDelete	This method deletes a node type property from the catalog.
	CatalogNodeTypePropertyGet	This method returns a node type property.
	CatalogNodeTypePropertyGetByName	This method returns a node type property providing its name.
	CatalogNodeTypePropertyUpdate	This method updates the node type property in the catalog.
	CatalogNodeTypeUpdate	This method updates the node type in the catalog.
	CatalogNodeTypeUploadBackground	This method uploads a new background for an existing node type.
	CatalogNodeTypeUploadImage	This method uploads a new image for an existing node type.
	CatalogVendorAdd	This method adds a new vendor to the catalog.
	CatalogVendorDelete	This method deletes a vendor from the catalog.
	CatalogVendorGet	This method gets a vendor from the catalog.
	CatalogVendorGetByName	This method gets a vendor by name.
	CatalogVendorUpdate	This method updates the specified vendor.
	DiagramGetHeight	This method gets the height of a diagram.
	DiagramGetLinksByTypeId	This method gets a list of link names providing the type and the diagram.
	DiagramGetMarginSize	This method gets the margin size of a diagram.

	Name	Description
	DiagramGetNodesByTypeGroup	This method gets a dictionary of node ids and names providing the type group and the diagram.
	DiagramGetNodesByTypeId	Gets a list of nodes of a selected type on a selected diagram.
	DiagramGetWidth	This method gets the width of a diagram.
	InstanceMoveToFront	This method moves an instance on a diagram to the front.
	InstanceSendToBack	This method sends an instance on a diagram to the back.
	LinkDelete	This method deletes a link.
	LinkGetPropertyValue	This method gets the value of a link property.
	LinkGetPropertyValueByName	This method gets the value of a link property passing the property name.
	LinkGetTypeId	This method gets the type id of a link.
	LinkInsert	This method inserts a link with a specific name and of a specific type providing the end nodes.
	LinkPropertyUpdate	This method updates a link property.
	LinkTypeGetId	This method gets the id of the link type providing its name.
	NodeDelete	This method deletes a node.
	NodeGetPropertyValue	This method gets the value of a node property.
	NodeGetPropertyValueByName	This method gets the value of a node property by passing the property name.
	NodeGetTypeGroup	This method gets the type group of a node.
	NodeGetTypeId	This method gets the type id of a node.
	NodeInsert	This method inserts a node with a specific name and of a specific type providing the parent diagram.

	Name	Description
	NodePropertyUpdate	This method updates a node property.
	NodeReparent	This method moves a node to another diagram.
	NodeSetCanMove	This method updates the 'CanMove' setting of a visNode based on the node id and diagram id.
	NodeSetHeight	This method sets the height of a visNode based on the node id and diagram id.
	NodeSetWidth	This method sets the width of a visNode based on the node id and diagram id.
	NodeSetX	This method sets the X coordinate of the top-left corner of a visNode based on the node id and diagram id.
	NodeSetY	This method sets the Y coordinate of the top-left corner of a visNode based on the node id and diagram id.
	NodesGetByName	Gets a list of nodes with the specified name.
	NodeTypeGetId	This method gets the id of the node type providing its name.
	TestConnection	This method tests the opened channel between the server and the client. This method does nothing unless a problem is detected, in which case it throws a FaultException(TDetail) error, including the problem description.
	VisNodeSetAttribute	This method updates a set of attributes of a visNode based on the node id and diagram id.

4.1.3 See Also

NetTerrain.WebApi Namespace

4.2 AdminGroupAdd Method

This method adds a new user group.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.2.1 Syntax

C#

```
long AdminGroupAdd(
  string name,
  Roles role
)
```

4.2.1.1 Parameters

name
Type: System.String
Group name.

role
Type: NetTerrain.WebApi.Roles
The role of the group.

4.2.1.2 Return Value

Type: **Int64** Returns the id of the newly created group.

4.2.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.2.3 Examples

The following example shows how to add a new user group using the AdminGroupAdd method.

```
try
{
    long groupId = api.AdminGroupAdd("Team Lumbergh",
Roles.PowerUser);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot add group: {0}", info.Detail.Details));
  }
```

4.2.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.3 AdminGroupDelete Method

This method deletes a user group.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.3.1 Syntax

C#

```
void AdminGroupDelete(
  long groupId
)
```

4.3.1.1 Parameters

groupId
Type: System.Int64
The id of the group to delete.

4.3.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.3.3 Remarks

Only non-system groups with no users can be deleted.

4.3.4 Examples

The following example shows how to delete a user group using the AdminGroupDelete method.

```
try
{
  // Get existing group.
  Group group = api.AdminGroupGetByName("Team Lumbergh");

  // Delete the group.
  api.AdminGroupDelete(group.Id);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
    "Cannot delete group: {0}", info.Detail.Details)
  );
}
```

4.3.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.4 AdminGroupGet Method

This method gets a user group.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.4.1 Syntax

C#

```
Group AdminGroupGet(
  long groupId
)
```

4.4.1.1 Parameters

groupId
Type: System.Int64
Group id.

4.4.1.2 Return Value

Type: Group Returns the group object.

4.4.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.4.3 Examples

The following example shows how to get a user group using the AdminGroupGet method.

```
try
{
  Group group = api.AdminGroupGet(12000000000018);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(
    string.Format( "Cannot get user group: {0}", info.Detail.Details)
    );
}
```

4.4.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.5 AdminGroupGetByName Method

This method gets a user group by name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.5.1 Syntax

C#

```
Group AdminGroupGetByName(
  string groupName
)
```

4.5.1.1 Parameters

groupName
Type: System.String
Group name.

4.5.1.2 Return Value

Type: Group Returns the group object.

4.5.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.5.3 Examples

The following example shows how to get a user group using the AdminGroupGetByName method.

```
try
{
  Group group = api.AdminGroupGetByName("Team Lumbergh");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(
    string.Format(  "Cannot get user group: {0}", info.Detail.Details)
  );
}
```

4.5.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.6 AdminGroupUpdate Method

This method updates the user group.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.6.1 Syntax

C#

```
void AdminGroupUpdate(
  Group group
)
```

4.6.1.1 Parameters

group
Type: NetTerrain.WebApi.Group
Group object with updated data.

4.6.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.6.3 Remarks

To update the user group get the group object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

System groups cannot be updated. A non-system group cannot be changed to a system group by setting the IsSystem parameter to true.

When altering the Name parameter (renaming the group) the new name must be unique throughout all groups.

4.6.4 Examples

The following example shows how to update the user group using the AdminGroupUpdate method.

```
try
{
  // Get node type object.
```

```
    Group group = api.AdminGroupGetByName("Team Lumbergh");

    // Update group parameters.
    group.Name = "Updated group name";
    group.Role = Roles.Editor;

    // Update the group.
    api.AdminGroupUpdate(group);
  }
  catch (FaultException<FaultInfo> info)
  {
    Console.WriteLine(string.Format(
        "Cannot update the user group: {0}", info.Detail.Details)
    );
  }
```

4.6.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.7 AdminUserAdd Method

This method adds a new user.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.7.1 Syntax

C#

```
long AdminUserAdd(
  string name,
  bool isAdAccount,
  bool overrideAdGroup,
  long groupId,
  string password,
  string email,
  string description,
  string comments )
```

4.7.1.1 Parameters

name
Type: System.String
User name.

isAdAccount
Type: System.Boolean
If set to true indicates that the user belongs to an Active Directory domain.

overrideAdGroup
Type: System.Boolean
If set to true forces an Active Directory user to use a manually specified Group instead of the one determined by the domain controller. This parameter is ignored for non-Active Directory users.

groupId
Type: System.Int64
Id of the user group. Should always be specified for native users. Ignored for Active Directory users when overrideAdGroup parameter is set to false.

password
Type: System.String
User password. Ignored for Active Directory users.

email
Type: System.String
User email. Ignored for Active Directory users.

description
Type: System.String
User description. Ignored for Active Directory users.

comments
Type: System.String
Additional comments. Ignored for Active Directory users.

4.7.1.2 Return Value

Type: **Int64** Returns the id of the newly created user.

4.7.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.7.3 Remarks

When adding an Active Directory user this method requires the name of the user to already exist in the Active Directory domain. New users cannot be added into an Active Directory domain using this method.

4.7.4 Examples

The following example shows how to add a new user using the AdminUserAdd method.

```
try
{
  // Get group for a new native user.
  Group powerUserGroup = api.AdminGroupGetByName("PowerUser");

  // Add a native user.
  long nativeUserId = api.AdminUserAdd(
      "Milton Waddams", false, false, powerUserGroup.Id,
"kjlGleFds$4",
      "Milton@initech.com", "This is a test native user", "");

  // Add an Active Directory user.
  long adUserId = api.AdminUserAdd(
      "MiltonADUser", true, false, 0, "", "", "", "");

  // Add an Active Directory user with group override.
  long adUserId2 = api.AdminUserAdd(
      "WaddamsADUser2", true, true, powerUserGroup.Id, "", "", "",
"");
}
catch (FaultException<FaultInfo> info)
```

```
{
  Console.WriteLine(string.Format(
    "Cannot add user: {0}", info.Detail.Details)
  );
}
```

4.7.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.8 AdminUserDelete Method

This method deletes a user.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.8.1 Syntax

C#

```
void AdminUserDelete(
  long userId
)
```

4.8.1.1 Parameters

userId
Type: System.Int64
The id of the user to delete.

4.8.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.8.3 Remarks

Service account used to open this WebApi channel cannot be deleted this way.

4.8.4 Examples

The following example shows how to delete a user using the AdminUserDelete method.

```
try
{
  // Get existing user.
  User user = api.AdminUserGetByName("Nina");

  // Delete the user.
  api.AdminUserDelete(user.Id);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
          "Cannot delete user: {0}", info.Detail.Details)
  );
}
```

4.8.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.9 AdminUserGet Method

This method gets a user.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.9.1 Syntax

C#

```
User AdminUserGet(
  long userId
)
```

4.9.1.1 Parameters

userId
Type: System.Int64
User id.

4.9.1.2 Return Value

Type: User Returns the user object.

4.9.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.9.3 Examples

The following example shows how to get a user using the AdminUserGet method.

```
try
{
  User user = api.AdminUserGet(94000000000132);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get user: {0}", info.Detail.Details)
  );
}
```

4.9.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.10 AdminUserGetByName Method

This method gets a user by name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.10.1 Syntax

C#

```
User AdminUserGetByName(
  string userName
)
```

4.10.1.1 Parameters

userName
Type: System.String
User name.

4.10.1.2 Return Value

Type: User Returns the user object.

4.10.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.10.3 Examples

The following example shows how to get a user using the AdminUserGetByName method.

```
try
{
  User user = api.AdminUserGetByName("Samir Nagheenanajar");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get user: {0}", info.Detail.Details)
  );
}
```

4.10.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.11 AdminUserSetPassword Method

This method changes a user password.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.11.1 Syntax

C#

```
void AdminUserSetPassword(
  long userId,
  string password
)
```

4.11.1.1 Parameters

userId
Type: System.Int64
User id.

password
Type: System.String
New user password.

4.11.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.11.3 Remarks

Password of current service account cannot be changed.

4.11.4 Examples

The following example shows how to change a user password using the AdminUserSetPassword method.

```
try
{
  User user = api.AdminUserGetByName("Michael Bolton");
  api.AdminUserSetPassword(user.Id, "klEslDS;3s");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get user: {0}", info.Detail.Details)
  );
}
```

4.11.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.12 AdminUserUpdate Method

This method updates the user.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.12.1 Syntax

C#

```
void AdminUserUpdate(
  User user
)
```

4.12.1.1 Parameters

user
Type: NetTerrain.WebApi.User
User object with updated data.

4.12.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.12.3 Remarks

To update the user get the user object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

User parameters of an Active Directory user stored on the Domain Acontroller cannot be updated.

User Name cannot be changed (user renaming is not possible).

Active Directory status of the user cannot be changed.

For Active Directory users the user group can only be changed if the IsOverrideAdGroup parameter is true.

User 'admin' account and current service user accounts cannot be changed.

4.12.4 Examples

The following example shows how to update the user using the AdminUserUpdate method.

```
try
{
  // Get node type object.
  User user = api.AdminUserGetByName("Bob Slydell");

  // Get group for user update.
  Group editorGroup = api.AdminGroupGetByName("Edit");

  // Update user parameters.
  user.Description = "I am a consultant";
  user.Comments = "Is this good for the company?";
  user.GroupId = editorGroup.Id;
  user.IsLocked = false;

  // Update the user.
  api.AdminUserUpdate(user);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot update the user: {0}", info.Detail.Details)
  );
}
```

4.12.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.13 CatalogAddLinkType Method

This method adds a new link type to the catalog. This is an older version of the method. It is adviced to use the newer method CatalogLinkTypeAdd(String, Boolean)

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.13.1 Syntax

C#

```
long CatalogAddLinkType(
  string name,
  bool isFavorite,
  float thickness,
  LinkStyles linkStyle
)
```

4.13.1.1 Parameters

name
Type: System.String
Name of the new type. This name must be unique throughout the link type catalog.

isFavorite
Type: System.Boolean
Flag to set the type as a favorite.

thickness
Type: System.Single
Link thickness.

linkStyle
Type: NetTerrain.WebApi.LinkStyles
Link style value of LinkStyles enumeration.

4.13.1.2 Return Value

Type: **Int64** Returns the id of the newly created link type.

4.13.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.13.3 Remarks

Link type attributes not specified in the method signature use the following default values:

SnappedToEdge	false
Color	#000000
CategoryId	null
StartArrow	null
EndArrow	null

4.13.4 Examples

The following example shows how to create a new link type using the CatalogAddLinkType method.

```
try
{
  long catalogLinkTypeId = api.CatalogAddLinkType(
    "New Link Type", true, 2, LinkStyles.DashDot);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Cannot create link type: {0}",

    info.Detail.Details)
  );
}
```

4.13.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.14 CatalogAddLinkTypeProperty Method

This method adds a new property to an existing link type in the catalog. This is an older copy of CatalogLinkTypePropertyAdd(Int64, String, String, Boolean, Boolean, Boolean) method. The new method was added to follow method naming convention.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.14.1 Syntax

C#

```
long CatalogAddLinkTypeProperty(
  long linkTypeId,
  string propertyName,
  string defaultValue,
  bool isMandatory,
  bool isDisplayed,
  bool isInProperties
)
```

4.14.1.1 Parameters

linkTypeId
Type: System.Int64
Target link type id.

propertyName
Type: System.String
Name of the new property. Must be unique for properties of this type.

defaultValue
Type: System.String
Default property value.

isMandatory
Type: System.Boolean
Makes the property mandatory if set to true.

isDisplayed
Type: System.Boolean
Makes the property displayed if set to true.

isInProperties
Type: System.Boolean
If set to true, the new property will be displayed in the properties list when a link of this type is selected in the netTerrain project.

4.14.1.2 Return Value

Type: **Int64** Returns the id of the created link type property.

4.14.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.14.3 Remarks

Property attributes not specified in the method signature use the following default values:

FontSize	10
FontColor	#000000
FillColor	null

4.14.4 Examples

The following example shows how to add a new link type property using the CatalogAddLinkTypeProperty method.

```
try
{
  long catalogLinkTypePropertyId = api.CatalogAddLinkTypeProperty(
       27000000000164, "Bandwidth", "defValue",
       true, true, true);
}
```

```
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot create link type property: {0}",
        info.Detail.Details)
  );
}
```

4.14.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.15 CatalogAddNodeType Method

This method adds a new node type to the catalog. This is an older version of the method. It is adviced to use the newer method CatalogNodeTypeAdd(String, NodeTypeGroups, String, Boolean) with better handling of the TypeGroupNumber parameter.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.15.1 Syntax

C#

```
void CatalogAddNodeType(
  NewNodeTypeInfo parameters
)
```

4.15.1.1 Parameters

parameters
Type: NetTerrain.WebApi.NewNodeTypeInfo
Parameters of a new node type.

4.15.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.15.3 Remarks

A new node type requires an associated icon image. This image cannot be passed to the server as a usual method parameter; instead it requires a file uploading process. As such, due to the specifics of the WCF service, this method cannot return the id of the created node type. You can retrieve this id in a subsequent method call by using NodeTypeGetId(String) or CatalogNodeTypeGetByName(String).

This method is an older version of the node type addition method and was kept for compatibility purposes. It is adviced to use the newer method CatalogNodeTypeAdd(String, NodeTypeGroups, String, Boolean) which sets the node type group based on the NodeTypeGroups enumeration rather than the TypeGroupNumber int value.

4.15.4 Examples

The following example shows how to add a new node type using the CatalogAddNodeType method.

```
try
{
  FileStream stream = new FileStream(
      @"c:\CiscoSwitchImage.png", FileMode.Open, FileAccess.Read);
      api.CatalogAddNodeType(new NewNodeTypeInfo  {
        ImageFileName = " CiscoSwitchImage.png",
        ImageByteStream = stream,
        Name = "Cisco Switch",
        TypeGroupNumber = 1,
        IsEnabled = true,
        IsFavourite = true,
        DefaultHeight = (float)0.5,
        DefaultWidth = (float)0.5
        });

  long typeId = api.NodeTypeGetId("Cisco Switch");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot create node type: {0}", info.Detail.Details)
  );
}
```

4.15.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.16 CatalogAddNodeTypeProperty Method

This method adds a new property to an existing node type in the catalog. This is an older copy of the CatalogNodeTypePropertyAdd(Int64, String, String, Boolean, Boolean, Boolean) method. The new method was added to follow better method naming convention.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.16.1 Syntax

C#

```
long CatalogAddNodeTypeProperty(
  long nodeTypeId,
  string propertyName,
  string defaultValue,
  bool isMandatory,
  bool isDisplayed,
  bool isInProperties
)
```

4.16.1.1 Parameters

nodeTypeId
Type: System.Int64
The node type id.

propertyName
Type: System.String
Name of the new property. Must be unique for properties of this type.

defaultValue
Type: System.String
Default property value.

isMandatory
Type: System.Boolean
Makes property mandatory if set to true.

isDisplayed
Type: System.Boolean
Makes property displayed if set to true.

isInProperties
Type: System.Boolean
If set to true the new property will be displayed in the properties list when a node of this type is selected on a diagram.

4.16.1.2 Return Value

Type: **Int64** Returns the id of the created node type property.

4.16.2 Exceptions

Exception	**Condition**
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.16.3 Remarks

Property attributes not specified in the method signature use the following default values:

FontSize	10
FontColor	#000000
FillColor	transparent

4.16.4 Examples

The following example shows how to add a new node type property using the CatalogAddNodeTypeProperty method.

```
try
{
  long nodeTypePropertyId = api.CatalogAddNodeTypeProperty(
26000000003247, "IP Address", "127.0.0.0",  true, true, true);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot create node type property: {0}", info.Detail.Details)
  );
}
```

4.16.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.17 CatalogGetLinkTypeProperties Method

This method gets a list of properties for a link type.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.17.1 Syntax

C#

```
Dictionary<long, string> CatalogGetLinkTypeProperties(
  long linkTypeId
)
```

4.17.1.1 Parameters

linkTypeId
Type: System.Int64
Link type id.

4.17.1.2 Return Value

Type: **Dictionary**(**Int64**, **String**) Returns a dictionary containing 'property id' - 'property name' pairs.

4.17.2 Exceptions

Exception	**Condition**
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.17.3 Remarks

The returned list will at a minimum contain the built-in 'Name' property.

4.17.4 Examples

The following example shows how to retrieve all the properties for the specified link type using the CatalogGetLinkTypeProperties method.

```
try
{
  Dictionary<long, string> list =
api.CatalogGetLinkTypeProperties(27000000000138);
    foreach (KeyValuePair<long, string> entry in list)  {
      Console.WriteLine(string.Format("Property id: {0}, name: {1}",
      entry.Key, entry.Value));
    }
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Cannot get properties list: {0}",
      info.Detail.Details)
    );
}
// The example will return an output such as the following:
// Property id: 29000000039974, name: A port
// Property id: 29000000039973, name: IP Address
// Property id: 29000000039971, name: Name
// Property id: 29000000039972, name: Zippity port
```

4.17.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.18 CatalogGetNodeTypeProperties Method

This method gets a list of properties for a node type.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.18.1 Syntax

C#

```
Dictionary<long, string> CatalogGetNodeTypeProperties(
  long nodeTypeId
)
```

4.18.1.1 Parameters

nodeTypeId
Type: System.Int64
Node type id.

4.18.1.2 Return Value

Type: **Dictionary**(**Int64**, **String**) Returns a dictionary containing 'property id' - 'property name' pairs.

4.18.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.18.3 Remarks

The returned list will at a minimum contain the built-in 'Name' property.

4.18.4 Examples

The following example shows how to retrieve all the properties for the specified node type using the CatalogGetNodeTypeProperties method.

```
try
{
  Dictionary<long, string> list =
api.CatalogGetNodeTypeProperties(26000000004197);
  foreach (KeyValuePair<long, string> entry in list)  {
      Console.WriteLine(string.Format("Property id: {0}, name: {1}",
        entry.Key, entry.Value));
  }
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Cannot get properties list: {0}",
        info.Detail.Details)
  );
}
// The example outputs something like the following:
// Property id: 28000000006473, name: Address
// Property id: 28000000006472, name: Phone No
// Property id: 28000000006470, name: Name
// Property id: 28000000006471, name: Shoe Size
```

4.18.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.19 CatalogLinkCategoryAdd Method

This method adds a new link category.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.19.1 Syntax

C#

```
long CatalogLinkCategoryAdd(
  string name,
  long parentId,
  string imageFileName,
  bool isFavorite
)
```

4.19.1.1 Parameters

name
Type: System.String
Link category name. This name must be unique throughout the link categories catalog.

parentId
Type: System.Int64
Parent category id. '0' should be used if no parent category required.

imageFileName
Type: System.String
Category icon image file name if the image has already been uploaded to the netTerrain application server. Must be empty otherwise.

isFavorite
Type: System.Boolean
Flag to set the category as a favorite.

4.19.1.2 Return Value

Type: **Int64** Returns the id of the newly created link category.

4.19.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.19.3 Remarks

To set the category icon image, it should first be uploaded to the netTerrain application separately using the CatalogLinkCategoryUploadImage(FileUploadAttributes) method (see the example). The same image can be used for different categories.

4.19.4 Examples

The following example shows how to add a new link category using the CatalogLinkCategoryAdd method.

```
try
{
  //Add a new category with a parent category (should exist) and
  // default icon image.
  long newCategoryId = api.CatalogLinkCategoryAdd( "Fiber", 0,
string.Empty, true);

  // Create file stream for image upload.
  FileStream stream = new FileStream(
      @"c:\Fiber.jpg", FileMode.Open, FileAccess.Read);

  // Upload the image for the new categpry.
  api.CatalogLinkCategoryUploadImage(new FileUploadAttributes() {
    FileName = @"Fiber.jpg", FileByteStream = stream, ObjectId =
newCategoryId });

  // Release the stream.
  stream.Dispose();
}
catch (FaultException<FaultInfo> info)
```

```
{
  Console.WriteLine(string.Format(
         "Cannot create link category: {0}",
         info.Detail.Details)
  );
}
```

4.19.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.20 CatalogLinkCategoryDelete Method

This method deletes a link category from the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.20.1 Syntax

C#

```
void CatalogLinkCategoryDelete(
  long linkCategoryId
)
```

4.20.1.1 Parameters

linkCategoryId
Type: System.Int64
Id of the link category to delete.

4.20.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.20.3 Remarks

A link category cannot be deleted if it has any associated link types or any child link categories.

4.20.4 Examples

The following example shows how to delete a category using the CatalogLinkCategoryDelete method.

```
try
{
  // Get existing link category.
  LinkCategory linkCategory =
api.CatalogLinkCategoryGetByName("Fiber");

  // Delete the link category.
  api.CatalogLinkCategoryDelete(linkCategory.Id);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot delete link category: {0}", info.Detail.Details)
  );
}
```

4.20.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.21 CatalogLinkCategoryGet Method

This method returns a link category.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.21.1 Syntax

C#

```
LinkCategory CatalogLinkCategoryGet(
  long linkCategoryId
)
```

4.21.1.1 Parameters

linkCategoryId
Type: System.Int64
Link category id.

4.21.1.2 Return Value

Type: LinkCategory Returns the link category object.

4.21.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.21.3 Examples

The following example shows how to get a link category using the CatalogLinkCategoryGet method.

```
try
{
  LinkCategory linkCategory =
api.CatalogLinkCategoryGet(32000000000351);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get link category: {0}",
        info.Detail.Details)
```

```
    );
}
```

4.21.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.22 CatalogLinkCategoryGetByName Method

This method returns a link category providing its name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.22.1 Syntax

C#

```
LinkCategory CatalogLinkCategoryGetByName(
  string linkCategoryName
)
```

4.22.1.1 Parameters

linkCategoryName
Type: System.String
Link category name.

4.22.1.2 Return Value

Type: LinkCategory Returns the link category object.

4.22.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.22.3 Examples

The following example shows how to get a link category using the CatalogLinkCategoryGetByName method.

```
try
{
  LinkCategory linkCategory =
api.CatalogLinkCategoryGetByName("Fiber");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get link category: {0}", info.Detail.Details)
  );
}
```

4.22.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.23 CatalogLinkCategoryUpdate Method

This method updates the link category in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.23.1 Syntax

C#

```
void CatalogLinkCategoryUpdate(
  LinkCategory linkCategory
)
```

4.23.1.1 Parameters

linkCategory
Type: NetTerrain.WebApi.LinkCategory
Link category object with updated data.

4.23.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.23.3 Remarks

To update the link category get the link category object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

The type group of the link category cannot be changed.

When altering the Name parameter (renaming the link category) the new name must be unique throughout all link categories.

When changing the ImageFileName parameter, the image file should already exist on the server. The same icon image can be used for different categories. To update the link category icon image with a new image file the CatalogLinkCategoryUploadImage(FileUploadAttributes) method should be used. Use an empty string to remove the icon image.

In case of setting or altering the ParentId parameter (reparenting) the link type group of the new parent link category must coincide with this link category. The new parent category should exist.

4.23.4 Examples

The following example shows how to update the link category using the CatalogLinkCategoryUpdate method.

```
try
{
  // Get link category object.
  LinkCategory linkCategory =
api.CatalogLinkCategoryGetByName("Fiber");

  // Update link category parameters.
  linkCategory.ImageFileName = "AnotherExistingIcon.jpg";
  linkCategory.IsFavorite = false;
  linkCategory.Name = "Renamed category";
  linkCategory.ParentId = 32000000000005;

  // Update the link category.
  api.CatalogLinkCategoryUpdate(linkCategory);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot update the link category: {0}", info.Detail.Details)
  );
}
```

4.23.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.24 CatalogLinkCategoryUploadImage Method

This method uploads a new icon image for an existing link category.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.24.1 Syntax

C#

```
void CatalogLinkCategoryUploadImage(
  FileUploadAttributes fileUploadAttributes
)
```

4.24.1.1 Parameters

fileUploadAttributes
Type: NetTerrain.WebApi.FileUploadAttributes
Attributes of a new link category icon image.

4.24.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.24.3 Examples

The following example shows how to upload an image using the CatalogLinkCategoryUploadImage method.

```
try
{
  // Get existing link category.
  LinkCategory linkCategoryToUpdate =
api.CatalogLinkCategoryGetByName("Fiber");
  FileStream stream = new FileStream(
      @"c:\Fiber.jpg", FileMode.Open, FileAccess.Read);
      api.CatalogLinkCategoryUploadImage(new FileUploadAttributes()
{
        FileName = "Fiber.jpg",
        FileByteStream = stream,
        ObjectId = linkCategoryToUpdate.Id
  });
}
```

```
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot upload link category image: {0}",
        info.Detail.Details)
  );
}
```

4.25 CatalogLinkOverrideAdd Method

This method adds a new override for the link type property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.25.1 Syntax

C#

```
long CatalogLinkOverrideAdd(
  long linkPropertyId,
  bool isOverride,
  OverrideRules rule,
  string listValue,
  string color,
  float thickness,
  LinkStyles linkStyle,
  ArrowStyles startArrow,
  ArrowStyles endArrow )
```

4.25.1.1 Parameters

linkPropertyId
Type: System.Int64
The id of the link type property.

isOverride
Type: System.Boolean
If set to true adds a real override, otherwise adds just a list value.

rule
Type: NetTerrain.WebApi.OverrideRules
Override rule used for this value.

listValue
Type: System.String
Value for the property's values list.

color
Type: System.String
Color hex code override for a link.

thickness
Type: System.Single
Thickness override for a link.

linkStyle
Type: NetTerrain.WebApi.LinkStyles
Line style override for a link.

startArrow
Type: NetTerrain.WebApi.ArrowStyles
Start arrow override for a link.

endArrow
Type: NetTerrain.WebApi.ArrowStyles
End arrow override for a link.

4.25.1.2 Return Value

Type: **Int64** Returns the id of the newly created link override object.

4.25.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.25.3 Examples

The following example shows how to add a new link override using the CatalogLinkOverrideAdd method.

```
try
{
  //Add a new link override '10'.
  long newOverrideId = api.CatalogLinkOverrideAdd(
      29000000000378,
      true,
      OverrideRules.GreaterThan,
      "10",
      "#123456",
      2.8f,
      LinkStyles.DashDot,
      ArrowStyles.Solid,
      ArrowStyles.SolidReversed
  );
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot create link override: {0}",
        info.Detail.Details)
  );
}
```

4.25.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.26 CatalogLinkOverrideAddListValue Method

This method adds a new list value for a link type property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.26.1 Syntax

C#

```
long CatalogLinkOverrideAddListValue(
  long linkPropertyId,
  string listValue
  )
```

4.26.1.1 Parameters

linkPropertyId
Type: System.Int64
The id of the link type property.

listValue
Type: System.String
Value for the property's values list.

4.26.1.2 Return Value

Type: **Int64** Returns the id of the newly created link override object representing this list value.

4.26.2 Exceptions

Exception	**Condition**
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.26.3 Remarks

A list value can also be viewed as an override entry that does not override anything.

4.26.4 Examples

The following example shows how to add a new link list value using the CatalogLinkOverrideAddListValue method.

```
try
{
  // Add a new list value '10'.
  long newOverrideId = api.CatalogLinkOverrideAddListValue(
      29000000000378, "10"
  );
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot create link list value: {0}",
        info.Detail.Details)
  );
}
```

4.26.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.27 CatalogLinkOverrideDelete Method

This method deletes a link override (list value) from the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.27.1 Syntax

C#

```
void CatalogLinkOverrideDelete(
  long overrideId
)
```

4.27.1.1 Parameters

overrideId
Type: System.Int64
Id of a link override to delete.

4.27.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.27.3 Remarks

A link override (or a list value) can be deleted even if it is currently active for some links.

4.27.4 Examples

The following example shows how to delete an override using the CatalogLinkOverrideDelete method.

```
try
{
  // Delete the link override.
  api.CatalogLinkOverrideDelete(142);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
         "Cannot delete link override: {0}",
         info.Detail.Details)
  );
}
```

4.27.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.28 CatalogLinkOverrideGet Method

This method returns a link override.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.28.1 Syntax

C#

```
LinkOverride CatalogLinkOverrideGet(
  long overrideId
)
```

4.28.1.1 Parameters

overrideId
Type: System.Int64
Link override id.

4.28.1.2 Return Value

Type: LinkOverride Returns the link override object.

4.28.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.28.3 Examples

The following example shows how to get a link override using the CatalogLinkOverrideGet method.

```
try
{
  LinkOverride linkOverride = api.CatalogLinkOverrideGet(142);
}
```

```
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get link override: {0}",
        info.Detail.Details)
  );
}
```

4.28.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.29 CatalogLinkOverrideGetByListValue Method

This method returns a link override providing its list value and link property id.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.29.1 Syntax

C#

```
LinkOverride CatalogLinkOverrideGetByListValue(
  long propertyId,
  string value
)
```

4.29.1.1 Parameters

propertyId
Type: System.Int64
Link property id.

value
Type: System.String
List value for this property.

4.29.1.2 Return Value

Type: LinkOverride Returns the link override object.

4.29.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.29.3 Examples

The following example shows how to get a link override using the CatalogLinkOverrideGetByListValue method.

```
try
{
  LinkOverride linkOverride =
api.CatalogLinkOverrideGetByListValue(29000000000378, "14");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get link override: {0}",
        info.Detail.Details)
  );
}
```

4.30 CatalogLinkOverridesGetByPropertyId Method

Gets a list of link overrides for a specified link type property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.30.1 Syntax

C#

```
List<long> CatalogLinkOverridesGetByPropertyId(
  long linkPropertyId
)
```

4.30.1.1 Parameters

linkPropertyId
Type: System.Int64
Link property id.

4.30.1.2 Return Value

Type: **List**(**Int64**) Returns a list of override id corresponding to the specified link property.

4.30.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.30.3 Examples

The following example shows how to get the list of link override ids using the CatalogLinkOverridesGetByPropertyId method.

```
try
{
  long[] linkOverrideIds =
api.CatalogLinkOverridesGetByPropertyId(29000000000378);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
    info.Detail.Details));
  }
```

4.30.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.31 CatalogLinkOverridesGetByTypeId Method

Gets a list of link overrides for all properties of a specified link type.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.31.1 Syntax

C#

```
List<long> CatalogLinkOverridesGetByTypeId(
  long linkTypeId
)
```

4.31.1.1 Parameters

linkTypeId
Type: System.Int64
Link type id.

4.31.1.2 Return Value

Type: **List**(**Int64**) Returns a list of override ids corresponding to properies belonging to the specified link type.

4.31.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.31.3 Examples

The following example shows how to get the list of link override ids using the CatalogLinkOverridesGetByTypeId method.

```
try
{
  long[] linkOverrideIds =
api.CatalogLinkOverridesGetByTypeId(27000000000023);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
  }
```

4.31.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.32 CatalogLinkOverrideUpdate Method

This method updates the link override in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.32.1 Syntax

C#

```
void CatalogLinkOverrideUpdate(
  LinkOverride linkOverride
)
```

4.32.1.1 Parameters

linkOverride
Type: NetTerrain.WebApi.LinkOverride
Link override object with updated data.

4.32.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.32.3 Remarks

To update the link override get the override object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

The link property id cannot be changed.

By altering the IsOverride parameter an override can be transformed into a list value and vice versa.

4.32.4 Examples

The following example shows how to update the link override using the CatalogLinkOverrideUpdate method.

```
try
{
  // Get link override object.
  LinkOverride linkOverride = api.CatalogLinkOverrideGet(143);
```

```
    // Update link override parameters.
    linkOverride.IsOverride = true;
    linkOverride.Value = "17";
    linkOverride.Rule = OverrideRules.LowerThan;
    linkOverride.Color = "#111222";
    linkOverride.Thickness = 2;
    linkOverride.LinkStyle = LinkStyles.DashDotDot;
    linkOverride.StartArrow = ArrowStyles.Diamond;
    linkOverride.EndArrow = ArrowStyles.Diamond;

    // Update the link override.
    api.CatalogLinkOverrideUpdate(linkOverride);
  }
  catch (FaultException<FaultInfo> info)
  {
    Console.WriteLine(string.Format(
          "Cannot update the link override: {0}", info.Detail.Details)
      );
    }
```

4.32.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.33 CatalogLinkTypeAdd Method

This method adds a new link type to the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.33.1 Syntax

C#

```
long CatalogLinkTypeAdd(
  string name,
  bool isFavorite
)
```

4.33.1.1 Parameters

name
Type: System.String
Link type name. This name must be unique throughout the link types catalog.

isFavorite
Type: System.Boolean
Flag to set the link type as a favorite.

4.33.1.2 Return Value

Type: **Int64** Returns the id of the newly created link type.

4.33.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.33.3 Examples

The following example shows how to add a new link type using the CatalogLinkTypeAdd method.

```
try
{
  //Add a new link type.
  long newLinkTypeId = api.CatalogLinkTypeAdd("SONET", true);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
         "Cannot create link type: {0}", info.Detail.Details)
  );
}
```

4.34 CatalogLinkTypeClone Method

This method clones an existing link type in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.34.1 Syntax

C#

```
long CatalogLinkTypeClone(
  long linkTypeId,
  string clonedTypeName
)
```

4.34.1.1 Parameters

linkTypeId
Type: System.Int64
Id of the source link type.

clonedTypeName
Type: System.String
Name of the cloned link type.

4.34.1.2 Return Value

Type: **Int64** Returns the cloned link type id.

4.34.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.34.3 Remarks

Only non-system link types can be cloned.

4.34.4 Examples

The following example shows how to clone a link type using the CatalogLinkTypeClone method.

```
try
{
  // Get existing link type.
  LinkType sourceLinkType =  api.CatalogLinkTypeGetByName("MPLS");

  // Clone the link type and change the name to 'Layer 2'.
  long clonedLinkTypeId =
api.CatalogLinkTypeClone(sourceLinkType.Id, "Layer 2");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot clone link type: {0}", info.Detail.Details)
  );
}
```

4.34.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.35 CatalogLinkTypeDelete Method

This method deletes a link type from the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.35.1 Syntax

C#

```
void CatalogLinkTypeDelete(
  long linkTypeId
)
```

4.35.1.1 Parameters

linkTypeId
Type: System.Int64
Id of a link type to delete.

4.35.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.35.3 Remarks

Only non-system link types without associated links can be deleted.

4.35.4 Examples

The following example shows how to delete a link type using the CatalogLinkTypeDelete method.

```
try
{
  // Get existing link type.
  LinkType linkType = api.CatalogLinkTypeGetByName("X.25");

  // Delete the link type.
  api.CatalogLinkTypeDelete(linkType.Id);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
         "Cannot delete link type: {0}", info.Detail.Details)
  );
}
```

4.35.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.36 CatalogLinkTypeGet Method

This method returns a link type.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.36.1 Syntax

C#

```
LinkType CatalogLinkTypeGet(
  long linkTypeId
)
```

4.36.1.1 Parameters

linkTypeId
Type: System.Int64
Link type id.

4.36.1.2 Return Value

Type: LinkType Returns the link type object.

4.36.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.36.3 Examples

The following example shows how to get a link type using the CatalogLinkTypeGet method.

```
try
{
  LinkType linkType = api.CatalogLinkTypeGet(26000000005732);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get link type: {0}", info.Detail.Details)
  );
}
```

4.36.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.37 CatalogLinkTypeGetByName Method

This method returns a link type providing its name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.37.1 Syntax

C#

```
LinkType CatalogLinkTypeGetByName(
  string linkTypeName
)
```

4.37.1.1 Parameters

linkTypeName
Type: System.String
Link type name.

4.37.1.2 Return Value

Type: LinkType Returns the link type object.

4.37.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.37.3 Examples

The following example shows how to get a link type using the CatalogLinkTypeGetByName method.

```
try
{
  LinkType linkType = api.CatalogLinkTypeGetByName("SONET");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get link type: {0}", info.Detail.Details)
  );
}
```

4.37.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.38 CatalogLinkTypePropertyAdd Method

This method adds a new property to an existing link type in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.38.1 Syntax

C#

```
long CatalogLinkTypePropertyAdd(
  long linkTypeId,
  string propertyName,
  string defaultValue,
  bool isMandatory,
  bool isDisplayed,
  bool isInProperties
)
```

4.38.1.1 Parameters

linkTypeId
Type: System.Int64
Target link type id.

propertyName
Type: System.String
Name of the new property. Must be unique for properties of this type.

defaultValue
Type: System.String
Default property value.

isMandatory
Type: System.Boolean
Makes the property mandatory if set to true.

isDisplayed
Type: System.Boolean
Makes the property displayed if set to true.

isInProperties
Type: System.Boolean
If set to true, the new property will be displayed in the properties list when a link of this type is selected in the netTerrain project.

4.38.1.2 Return Value

Type: **Int64** Returns the id of the created link type property.

4.38.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.38.3 Remarks

Property attributes not specified in the method signature use the following default values:

FontSize	10
FontColor	#000000
FillColor	null

4.38.4 Examples

The following example shows how to add a new link type property using the CatalogLinkTypePropertyAdd method.

```
try
{
  long catalogLinkTypePropertyId = api.CatalogLinkTypePropertyAdd(
      27000000000164, "Color", "Red",  true, true, true
  );
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
         "Cannot create link type property: {0}", info.Detail.Details)
```

```
    );
}
```

4.38.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.39 CatalogLinkTypePropertyDelete Method

This method deletes a link type property from the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.39.1 Syntax

C#

```
void CatalogLinkTypePropertyDelete(
  long linkTypePropertyId
)
```

4.39.1.1 Parameters

linkTypePropertyId
Type: System.Int64
Id of the link type property to delete.

4.39.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.39.3 Remarks

Only non-system link type properties can be deleted.

4.39.4 Examples

The following example shows how to delete a link type property using the CatalogLinkTypePropertyDelete method.

```
try
{
  // Get existing link type property.
  LinkTypeProperty linkTypeProperty =
api.CatalogLinkTypePropertyGetByName(
    "Color",  27000000000077
    );

  // Delete the link type property.
  api.CatalogLinkTypePropertyDelete(linkTypeProperty.Id);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot delete link type property: {0}", info.Detail.Details)
  );
}
```

4.40 CatalogLinkTypePropertyGet Method

This method returns a link type property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.40.1 Syntax

C#

```
LinkTypeProperty CatalogLinkTypePropertyGet(
  long linkTypePropertyId
)
```

4.40.1.1 Parameters

linkTypePropertyId
Type: System.Int64
Link type property id.

4.40.1.2 Return Value

Type: LinkTypeProperty Returns the link type property object.

4.40.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.40.3 Examples

The following example shows how to get a link type properties using the CatalogLinkTypePropertyGet method.

```
try
{
  LinkTypeProperty linkTypeProperty =
api.CatalogLinkTypePropertyGet(29000000000385);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get link type property: {0}", info.Detail.Details)
```

```
    );
}
```

4.40.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.41 CatalogLinkTypePropertyGetByName Method

This method returns a link type property providing its name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.41.1 Syntax

C#

```
LinkTypeProperty CatalogLinkTypePropertyGetByName(
  string name,
  long linkTypeId
)
```

4.41.1.1 Parameters

name
Type: System.String
Link type property name.

linkTypeId
Type: System.Int64

4.41.1.2 Return Value

Type: LinkTypeProperty Returns the link type property object.

4.41.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.41.3 Examples

The following example shows how to get a link type property using the CatalogLinkTypePropertyGetByName method.

```
try
{
  LinkTypeProperty linkTypeProperty =
api.CatalogLinkTypePropertyGetByName(
      "Color", 27000000000077
    );
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get link type property: {0}", info.Detail.Details)
  );
}
```

4.42 CatalogLinkTypePropertyUpdate Method

This method updates the link type property in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.42.1 Syntax

C#

```
void CatalogLinkTypePropertyUpdate(
  LinkTypeProperty linkTypeProperty
)
```

4.42.1.1 Parameters

linkTypeProperty
Type: NetTerrain.WebApi.LinkTypeProperty
Link type property object with updated data.

4.42.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.42.3 Remarks

To update the link type property get the link type property object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

The Link type id cannot be changed.

When altering the Name parameter (renaming the link type property) the new name must be unique. Only non-system properties can be renamed.

System status of a property cannot be changed. The following parameters of system properties cannot be changed: Name, IsInProperties, Mandatory, LockList, Position.

The IsTypeField parameter cannot be changed. For type fields the following parameters cannot be changed: DefaultValue, IsUniqueForThisType, IsUniqueForAllTypes.

The LockList parameter cannot be set to 'true' if a property does not have any list values.

4.42.4 Examples

The following example shows how to update the link type property using the CatalogLinkTypePropertyUpdate method.

```
try
{
  // Get link type property object.
  LinkTypeProperty linkTypeProperty =
api.CatalogLinkTypePropertyGet(29000000000385);
```

```
    // Update link type parameters.
    linkTypeProperty.Name = "Color";
    linkTypeProperty.Angle = 20;
    linkTypeProperty.Bold = true;
    linkTypeProperty.DefaultValue = "Red";
    linkTypeProperty.Displayed = true;
    linkTypeProperty.FillColor = "#123123";
    linkTypeProperty.FontColor = "#321321";
    linkTypeProperty.FontFamily = FontFamilies.LucidaConsole;
    linkTypeProperty.FontSize = 12;
    linkTypeProperty.IsInProperties = true;
    linkTypeProperty.IsUniqueForAllTypes = false;
    linkTypeProperty.IsUniqueForThisType = false;
    linkTypeProperty.Italic = false;
    linkTypeProperty.Justification = TextJustification.Center;
    linkTypeProperty.LockList = false;
    linkTypeProperty.OffsetX = 0;
    linkTypeProperty.OffsetY = 0;
    linkTypeProperty.Position = 3;
    linkTypeProperty.TextAlign = TextAligns.Center;
    linkTypeProperty.Underline = false;

    // Update the link type.
    api.CatalogLinkTypePropertyUpdate(linkTypeProperty);
  }
  catch (FaultException<FaultInfo> info)
  {
    Console.WriteLine(string.Format(
          "Cannot update the link type: {0}", info.Detail.Details)
    );
  }
```

4.42.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.43 CatalogLinkTypeUpdate Method

This method updates the link type in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.43.1 Syntax

C#

```
void CatalogLinkTypeUpdate(  LinkType linkType )
```

4.43.1.1 Parameters

linkType
Type: NetTerrain.WebApi.LinkType
Link type object with updated data.

4.43.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.43.3 Remarks

To update the link type get the link type object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

System link types cannot be updated. The IsSystem parameter cannot be altered either.

When altering the Name parameter (renaming the link type) the new name must be unique.

A favorite link type or a link type with existing link instances cannot be disabled.

4.43.4 Examples

The following example shows how to update the link type using the CatalogLinkTypeUpdate method.

```
try
{
  // Get link type object.
```

```
    LinkType linkType = api.CatalogLinkTypeGetByName("Fiber");

    // Update link type parameters.
    linkType.Name = "Copper";
    linkType.IsFavorite = true;
    linkType.CategoryId = 32000000000003;
    linkType.Color = "#123123";
    linkType.LinkStyle = LinkStyles.Dot;
    linkType.StartArrow = ArrowStyles.Open;
    linkType.EndArrow = ArrowStyles.OpenReversed;
    linkType.Thickness = 2.5f;
    linkType.IsSnappedToEdge = true;
    linkType.IsMatchingPortConnectors = false;

    // Update the link type.
    api.CatalogLinkTypeUpdate(linkType);
  }
  catch (FaultException<FaultInfo> info)
  {
    Console.WriteLine(string.Format(
          "Cannot update the link type: {0}", info.Detail.Details)
      );
  }
```

4.43.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.44 CatalogNodeCategoryAdd Method

This method adds a new node category.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.44.1 Syntax

C#

```
long CatalogNodeCategoryAdd(
  string name,
  NodeTypeGroups typeGroup,
  long parentId,
  string imageFileName,
  bool isFavorite )
```

4.44.1.1 Parameters

name
Type: System.String
Node category name. This name must be unique throughout the node categories catalog.

typeGroup
Type: NetTerrain.WebApi.NodeTypeGroups
Node type group value.

parentId
Type: System.Int64
Parent category id. '0' should be used if no parent category is required.

imageFileName
Type: System.String
Category icon image file name for an image already uploaded to the netTerrain application server. Should be empty in all other instances.

isFavorite
Type: System.Boolean
Flag to set the category as a favorite.

4.44.1.2 Return Value

Type: **Int64** Returns the id of the newly created node category.

4.44.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.44.3 Remarks

To set category icon image the image should be uploaded to the netTerrain application separately with the CatalogNodeCategoryUploadImage(FileUploadAttributes) method (see the example). Alternatively a previously uploaded category icon image can be used if its name is known. The same image can be used for different categories.

4.44.4 Examples

The following example shows how to add a new node category using the CatalogNodeCategoryAdd method.

```
try
{
  //Add a new category with a parent category (should exist) and
  // default icon image.
  long newCategoryId = api.CatalogNodeCategoryAdd(  "Switches",
NodeTypeGroups.Device,  0,  string.Empty,  true);

  // Create file stream for image upload.
  FileStream stream = new FileStream(
      @"c:\Switch.jpg", FileMode.Open, FileAccess.Read);

  // Upload the image for the new categpry.
    api.CatalogNodeCategoryUploadImage(new FileUploadAttributes()  {
      FileName = @"Switch.jpg",
      FileByteStream = stream,  ObjectId = newCategoryId  });

  // Release the stream.
  stream.Dispose();
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot create node category: {0}", info.Detail.Details)
    );
}
```

4.44.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.45 CatalogNodeCategoryDelete Method

This method deletes a node category from the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.45.1 Syntax

C#

```
void CatalogNodeCategoryDelete(
    long nodeCategoryId )
```

4.45.1.1 Parameters

nodeCategoryId
Type: System.Int64
If of a node category to delete.

4.45.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.45.3 Remarks

A node category cannot be deleted if it has any associated node type or any child node category.

4.45.4 Examples

The following example shows how to delete a category using the CatalogNodeCategoryDelete method.

```
try
{
  // Get existing node category.
  NodeCategory nodeCategory =
api.CatalogNodeCategoryGetByName("Switches");

  // Delete the node category.
  api.CatalogNodeCategoryDelete(nodeCategory.Id);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot delete node category: {0}", info.Detail.Details)
    );
}
```

4.45.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.46 CatalogNodeCategoryGet Method

This method returns a node category.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.46.1 Syntax

C#

```
NodeCategory CatalogNodeCategoryGet(
  long nodeCategoryId )
```

4.46.1.1 Parameters

nodeCategoryId
Type: System.Int64
Node category id.

4.46.1.2 Return Value

Type: NodeCategory Returns the node category object.

4.46.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.46.3 Examples

The following example shows how to get a node category using the CatalogNodeCategoryGet method.

```
try
{
  NodeCategory nodeCategory =
api.CatalogNodeCategoryGet(31000000000351);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get node category: {0}", info.Detail.Details)
    );
}
```

4.46.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.47 CatalogNodeCategoryGetByName Method

This method returns a node category providing its name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.47.1 Syntax

C#

```
NodeCategory CatalogNodeCategoryGetByName(
  string nodeCategoryName )
```

4.47.1.1 Parameters

nodeCategoryName
Type: System.String
Node category name.

4.47.1.2 Return Value

Type: NodeCategory Returns the node category object.

4.47.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.47.3 Examples

The following example shows how to get a node category using the CatalogNodeCategoryGetByName method.

```
try
{
  NodeCategory nodeCategory =
api.CatalogNodeCategoryGetByName("Switches");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
        "Cannot get node category: {0}", info.Detail.Details)
    );
}
```

4.47.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.48 CatalogNodeCategoryUpdate Method

This method updates the node category in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.48.1 Syntax

C#

```
void CatalogNodeCategoryUpdate(
  NodeCategory nodeCategory
)
```

4.48.1.1 Parameters

nodeCategory
Type: NetTerrain.WebApi.NodeCategory
Node category object with updated data.

4.48.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.48.3 Remarks

To update the node category get the node category object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

The type group of the node category cannot be changed.

When altering the Name parameter (renaming the node category) the new name must be unique in the catalog of node categories.

When changing the ImageFileName parameter the image file should already exist on the server. The same icon image can be used for different categories. To update the node category icon image with a new image file the CatalogNodeCategoryUploadImage(FileUploadAttributes) method should be used. Use an empty string to remove the icon image.

To set or alter the ParentId parameter (reparenting) the node type group of the new parent node category must coincide with this node category. The new parent category should exist.

4.48.4 Examples

The following example shows how to update the node category using the CatalogNodeCategoryUpdate method.

```
try
{
  // Get node category object.
  NodeCategory nodeCategory =
api.CatalogNodeCategoryGetByName("Cooling Equipment");
```

```
    // Update node category parameters.
    nodeCategory.ImageFileName = "Cooling.jpg";
    nodeCategory.IsFavorite = false;
    nodeCategory.Name = "Cooling";
    nodeCategory.ParentId = 31000000000005;

    // Update the node category.
    api.CatalogNodeCategoryUpdate(nodeCategory);
  }
  catch (FaultException<FaultInfo> info)
  {
    Console.WriteLine(string.Format(
        "Cannot update the node category: {0}", info.Detail.Details)
    );
  }
```

4.48.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.49 CatalogNodeCategoryUploadImage Method

This method uploads a new icon image for an existing node category.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.49.1 Syntax

C#

```
void CatalogNodeCategoryUploadImage(
  FileUploadAttributes fileUploadAttributes
)
```

4.49.1.1 Parameters

fileUploadAttributes
Type: NetTerrain.WebApi.FileUploadAttributes
Attributes of a new node category icon image.

4.49.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.49.3 Examples

The following example shows how to upload an image using the CatalogNodeCategoryUploadImage method.

```
try
{
  // Get existing node category.
  NodeCategory nodeCategoryToUpdate =
  api.CatalogNodeCategoryGetByName("Cooling");

  FileStream stream = new FileStream(
      @"c:\Cooling.jpg", FileMode.Open, FileAccess.Read);
  api.CatalogNodeCategoryUploadImage(new FileUploadAttributes()
  {
    FileName = "Cooling.jpg",
    FileByteStream = stream,
    ObjectId = nodeCategoryToUpdate.Id
    }
  );
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(

      "Cannot upload node category image: {0}", info.Detail.Details)
  );
}
```

4.50 CatalogNodeOverrideAdd Method

This method adds a new override for a node type property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.50.1 Syntax

C#

```
long CatalogNodeOverrideAdd(
  long nodePropertyId,
  bool isOverride,
  OverrideRules rule,
  string listValue,
  string imageFileName )
```

4.50.1.1 Parameters

nodePropertyId
Type: System.Int64
The id of the node type property.

isOverride
Type: System.Boolean
If set to true adds a real override, otherwise adds a list value.

rule
Type: NetTerrain.WebApi.OverrideRules
Override rule used for this value.

listValue
Type: System.String
Value for the property.

imageFileName
Type: System.String
Override icon image file name if the image has already been uploaded to netTerrain application. Should be empty in all other instances.

4.50.1.2 Return Value

Type: **Int64** Returns the id of the newly created node override object.

4.50.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.50.3 Remarks

To set the node override icon image the image should be uploaded to the netTerrain application separately with the CatalogNodeOverrideUploadImage(FileUploadAttributes) method (see the example). Alternatively a previously uploaded override image can be used if its name is known. The same image can be used for different overrides.

4.50.4 Examples

The following example shows how to add a new node override using the CatalogNodeOverrideAdd method.

```
try
{
  //Add a new node override '10'.
  long newOverrideId = api.CatalogNodeOverrideAdd(  28000000022124,
true, OverrideRules.GreaterThan, "10", "");

  // Create file stream for image upload.
  FileStream stream = new FileStream(
    @"c:\OverrideImage.jpg", FileMode.Open, FileAccess.Read);

  // Upload the image for the new override.
  api.CatalogNodeOverrideUploadImage(new FileUploadAttributes()  {
    FileName = @"OverrideImage.jpg",
    FileByteStream = stream,  ObjectId = newOverrideId
  });

  // Release the stream.
  stream.Dispose();
}
```

```
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot create node override: {0}", info.Detail.Details)
  );
}
```

4.50.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.51 CatalogNodeOverrideAddListValue Method

This method adds a new list value for a node type property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.51.1 Syntax

C#

```
long CatalogNodeOverrideAddListValue(
  long nodePropertyId,
  string listValue )
```

4.51.1.1 Parameters

nodePropertyId
Type: System.Int64
The id of the node type property.

listValue
Type: System.String
Value for the property.

4.51.1.2 Return Value

Type: **Int64** Returns the id of the newly created node override object representing this list value.

4.51.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.51.3 Remarks

A List value can be interpreted as an override item that does not override anything.

4.51.4 Examples

The following example shows how to add a new node list value using the CatalogNodeOverrideAddListValue method.

```
try
{
  // Add a new list value '10'.
  long newOverrideId = api.CatalogNodeOverrideAddListValue(
28000000022124, "10");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot create node list value: {0}", info.Detail.Details)
  );
}
```

4.51.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.52 CatalogNodeOverrideDelete Method

This method deletes a node override (list value) from the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.52.1 Syntax

C#

```
void CatalogNodeOverrideDelete(
  long overrideId )
```

4.52.1.1 Parameters

overrideId
Type: System.Int64
Id of a node override to delete.

4.52.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.52.3 Remarks

A node override (or a list value) can be deleted even if it is currently active for some nodes.

4.52.4 Examples

The following example shows how to delete an override using the CatalogNodeOverrideDelete method.

```
try
{
  // Delete the node override.
  api.CatalogNodeOverrideDelete(516);
}
```

```
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot delete node override: {0}", info.Detail.Details)
  );
}
```

4.52.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.53 CatalogNodeOverrideGet Method

This method returns a node override.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.53.1 Syntax

C#

```
NodeOverride CatalogNodeOverrideGet(
  long overrideId
)
```

4.53.1.1 Parameters

overrideId
Type: System.Int64
Node override id.

4.53.1.2 Return Value

Type: NodeOverride Returns the node override object.

4.53.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.53.3 Examples

The following example shows how to get a node override using the CatalogNodeOverrideGet method.

```
try
{
  NodeOverride nodeOverride = api.CatalogNodeOverrideGet(516);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get node override: {0}", info.Detail.Details)
  );
}
```

4.53.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.54 CatalogNodeOverrideGetByListValue Method

This method returns a node override providing its list value and node property id.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.54.1 Syntax

C#

```
NodeOverride CatalogNodeOverrideGetByListValue(
  long propertyId,
    string value
)
```

4.54.1.1 Parameters

propertyId
Type: System.Int64
Node property id.

value
Type: System.String
List value for this property.

4.54.1.2 Return Value

Type: NodeOverride Returns the node override object.

4.54.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.54.3 Examples

The following example shows how to get a node override using the CatalogNodeOverrideGetByListValue method.

```
try
{
  NodeOverride nodeOverride =
api.CatalogNodeOverrideGetByListValue(28000000022124, "10");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get node override: {0}", info.Detail.Details)
```

```
    );
}
```

4.55 CatalogNodeOverridesGetByPropertyId Method

Gets a list of node overrides for a specified node type property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.55.1 Syntax

C#

```
List<long> CatalogNodeOverridesGetByPropertyId(
  long nodePropertyId
)
```

4.55.1.1 Parameters

nodePropertyId
Type: System.Int64
Node property id.

4.55.1.2 Return Value

Type: **List**(**Int64**) Returns a list of override ids corresponding to the specified node property.

4.55.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.55.3 Examples

The following example shows how to get the list of node override ids using the CatalogNodeOverridesGetByPropertyId method.

```
try
{
  long[] nodeOverrideIds =
api.CatalogNodeOverridesGetByPropertyId(28000000022124);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.55.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.56 CatalogNodeOverridesGetByTypeId Method

Gets a list of node overrides for all properties of a specified node type.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.56.1 Syntax

C#

```
List<long> CatalogNodeOverridesGetByTypeId(
  long nodeTypeId
)
```

4.56.1.1 Parameters

nodeTypeId
Type: System.Int64
Node type id.

4.56.1.2 Return Value

Type: **List**(**Int64**) Returns a list of override ids corresponding to properties belonging to the specified node type.

4.56.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.56.3 Examples

The following example shows how to get the list of node override ids using the CatalogNodeOverridesGetByTypeId method.

```
try
{
  long[] nodeOverrideIds =
api.CatalogNodeOverridesGetByTypeId(26000000003376);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.56.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.57 CatalogNodeOverrideUpdate Method

This method updates a node override in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.57.1 Syntax

C#

```
void CatalogNodeOverrideUpdate(
  NodeOverride nodeOverride
)
```

4.57.1.1 Parameters

nodeOverride
Type: NetTerrain.WebApi.NodeOverride
Node override object with updated data.

4.57.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.57.3 Remarks

To update the node override get the override object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

The node property id cannot be changed.

By altering the IsOverride parameter an override can be changed to a list value and vice versa.

When changing the ImageFileName parameter the image file should already exist on the server. The same image can be used for different overrides. To update the node override image with a new image the file CatalogNodeOverrideUploadImage(FileUploadAttributes) method should be used.

4.57.4 Examples

The following example shows how to update the node override using the CatalogNodeOverrideUpdate method.

```
try
{
  // Get node override object.
  NodeOverride nodeOverride = api.CatalogNodeOverrideGet(516);

  // Update node override parameters.
  nodeOverride.ImageFileName = "anotherOverrideImage.png";
  nodeOverride.IsOverride = true;
  nodeOverride.Value = "12";
  nodeOverride.Rule = OverrideRules.Equals;
  nodeOverride.InstanceEffect =
InstanceEffects.GreenTriangleIndicator;
  nodeOverride.ParentEffect =
InstanceEffects.OrangeTriangleIndicator;
  nodeOverride.UpwardsPropagation = UpwardsPropagations.ParentOnly;

  // Update the node override.
  api.CatalogNodeOverrideUpdate(nodeOverride);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(
      string.Format(  "Cannot update the node override: {0}",
info.Detail.Details)
  );
}
```

4.57.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.58 CatalogNodeOverrideUploadImage Method

This method uploads a new override image for an existing node override.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.58.1 Syntax

C#

```
void CatalogNodeOverrideUploadImage(
  FileUploadAttributes fileUploadAttributes
)
```

4.58.1.1 Parameters

fileUploadAttributes
Type: NetTerrain.WebApi.FileUploadAttributes
Attributes of a new node override icon image.

4.58.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.58.3 Examples

The following example shows how to upload an image using the CatalogNodeOverrideUploadImage method.

```
try
{
  // Get existing node override.
  NodeOverride nodeOverrideToUpdate =
api.CatalogNodeOverrideGet(516);
  FileStream stream = new FileStream(
     @"c:\Green.jpg", FileMode.Open, FileAccess.Read);
```

```
      api.CatalogNodeOverrideUploadImage(
       new FileUploadAttributes()  {
       FileName = "Green.jpg",
       FileByteStream = stream,  ObjectId = nodeOverrideToUpdate.Id  }
       );
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot upload node override image: {0}", info.Detail.Details)
  );
}
```

4.59 CatalogNodeTypeAdd Method

This method adds a new node type to the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.59.1 Syntax

C#

```
long CatalogNodeTypeAdd(
  string name,
  NodeTypeGroups typeGroup,
  string imageFileName,
  bool isFavorite
)
```

4.59.1.1 Parameters

name
Type: System.String
Node type name. This name must be unique throughout the node types catalog.

typeGroup
Type: NetTerrain.WebApi.NodeTypeGroups
Node type group value.

imageFileName
Type: System.String
Node type icon image file name if the image has already been uploaded to netTerrain application server. Should be empty for all other instances.

isFavorite
Type: System.Boolean
Flag to set the node type as a favorite.

4.59.1.2 Return Value

Type: **Int64** Returns the id of the newly created node type.

4.59.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.59.3 Remarks

It is only possible to add node types of following type groups: Node, Device, Rack, Card.

4.59.4 Examples

The following example shows how to add a new node type using the CatalogNodeTypeAdd method.

```
try
{
  //Add a new node type.
  long newNodeTypeId = api.CatalogNodeTypeAdd(  "Buildings",
NodeTypeGroups.Device,  string.Empty,  true);

  // Create file stream for image upload.
  FileStream stream = new FileStream(
    @"c:\Building.jpg", FileMode.Open, FileAccess.Read);

  // Upload the image for the new categpry.
    api.CatalogNodeTypeUploadImage(new FileUploadAttributes1()  {
      FileName = @"Building.jpg",
      FileByteStream = stream,  ObjectId = newNodeTypeId
      }
    );

  // Release the stream.
  stream.Dispose();
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot create node type: {0}", info.Detail.Details)
  );
}
```

4.59.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.60 CatalogNodeTypeClone Method

This method clones an existing node type in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.60.1 Syntax

C#

```
long CatalogNodeTypeClone(
  long nodeTypeId,
  string clonedTypeName
)
```

4.60.1.1 Parameters

nodeTypeId
Type: System.Int64
Id of the source node type.

clonedTypeName
Type: System.String
Name of the cloned node type.

4.60.1.2 Return Value

Type: **Int64** Returns the cloned node type id.

4.60.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.60.3 Remarks

Only non-system node types can be cloned.

4.60.4 Examples

The following example shows how to clone a node type using the CatalogNodeTypeClone method.

```
try
{
  // Get existing node type.
  NodeType sourceNodeType =  api.CatalogNodeTypeGetByName("Floors");

  // Clone the node type and give it name 'Cloned Type'.
  long clonedNodeTypeId =
api.CatalogNodeTypeClone(sourceNodeType.Id, "Rooms");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot clone node type: {0}", info.Detail.Details)
  );
}
```

4.60.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.61 CatalogNodeTypeDelete Method

This method deletes a node type from the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.61.1 Syntax

C#

```
void CatalogNodeTypeDelete(
    long nodeTypeId
)
```

4.61.1.1 Parameters

nodeTypeId
Type: System.Int64
Id of a node type to delete.

4.61.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.61.3 Remarks

Only non-system node types without associated nodes can be deleted.

4.61.4 Examples

The following example shows how to delete a node type using the CatalogNodeTypeDelete method.

```
try
{
  // Get node type.
  NodeType nodeType = api.CatalogNodeTypeGetByName("Floors");

  // Delete the node type.
  api.CatalogNodeTypeDelete(nodeType.Id);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot delete node type: {0}", info.Detail.Details)
  );
}
```

4.61.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.62 CatalogNodeTypeGet Method

This method returns a node type.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.62.1 Syntax

C#

```
NodeType CatalogNodeTypeGet(
  long nodeTypeId
)
```

4.62.1.1 Parameters

nodeTypeId
Type: System.Int64
Node type id.

4.62.1.2 Return Value

Type: NodeType Returns the node type object.

4.62.2 Exceptions

Exception	**Condition**
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.62.3 Examples

The following example shows how to get a node type using the CatalogNodeTypeGet method.

```
try
{
  NodeType nodeType = api.CatalogNodeTypeGet(26000000005732);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get node type: {0}", info.Detail.Details)
  );
}
```

4.62.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.63 CatalogNodeTypeGetByName Method

This method returns a node type providing its name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.63.1 Syntax

C#

```
NodeType CatalogNodeTypeGetByName(
  string nodeTypeName
)
```

4.63.1.1 Parameters

nodeTypeName
Type: System.String
Node type name.

4.63.1.2 Return Value

Type: NodeType Returns the node type object.

4.63.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.63.3 Examples

The following example shows how to get a node type using the CatalogNodeTypeGetByName method.

```
try
{
  NodeType nodeType = api.CatalogNodeTypeGetByName("Towers");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get node type: {0}", info.Detail.Details)
  );
}
```

4.63.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.64 CatalogNodeTypePropertyAdd Method

This method adds a new property to an existing node type in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.64.1 Syntax

C#

```
long CatalogNodeTypePropertyAdd(
  long nodeTypeId,
  string propertyName,
  string defaultValue,
  bool isMandatory,
  bool isDisplayed,
  bool isInProperties )
```

4.64.1.1 Parameters

nodeTypeId
Type: System.Int64
The node type id.

propertyName
Type: System.String
The name of the new property. Must be unique for properties of this type.

defaultValue
Type: System.String
Default property value.

isMandatory
Type: System.Boolean
Makes the property mandatory if set to true.

isDisplayed
Type: System.Boolean
Makes the property displayed if set to true.

isInProperties
Type: System.Boolean
If set to true the new property will be displayed in the properties list when a node of this type is selected on a diagram.

4.64.1.2 Return Value

Type: **Int64** Returns the id of the created node type property.

4.64.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.64.3 Remarks

Property attributes not specified in the method signature use the following default values:

FontSize	10
FontColor	#000000
FillColor	transparent

4.64.4 Examples

The following example shows how to add a new node type property using the CatalogNodeTypePropertyAdd method.

```
try
{
  long nodeTypePropertyId = api.CatalogNodeTypePropertyAdd(
26000000003247, "Diameter", "10",  true, true, true);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot create node type property: {0}", info.Detail.Details)
  );
}
```

4.64.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.65 CatalogNodeTypePropertyDelete Method

This method deletes a node type property from the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.65.1 Syntax

C#

```
void CatalogNodeTypePropertyDelete(
  long nodeTypePropertyId
)
```

4.65.1.1 Parameters

nodeTypePropertyId
Type: System.Int64
Id of a node type property to delete.

4.65.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.65.3 Remarks

Only non-system node type properties can be deleted.

4.65.4 Examples

The following example shows how to delete a node type property using the CatalogNodeTypePropertyDelete method.

```
try
{
  // Get node type property.
  NodeTypeProperty nodeTypeProperty =
api.CatalogNodeTypePropertyGetByName(  "Depth",    26000000003382);

  // Delete the node type property.
  api.CatalogNodeTypePropertyDelete(nodeTypeProperty.Id);
}
```

```
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot delete node type property: {0}", info.Detail.Details)
  );
}
```

4.66 CatalogNodeTypePropertyGet Method

This method returns a node type property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.66.1 Syntax

C#

```
NodeTypeProperty CatalogNodeTypePropertyGet(
  long nodeTypePropertyId
)
```

4.66.1.1 Parameters

nodeTypePropertyId
Type: System.Int64
Node type property id.

4.66.1.2 Return Value

Type: NodeTypeProperty Returns the node type property object.

4.66.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.66.3 Examples

The following example shows how to get a node type properties using the CatalogNodeTypePropertyGet method.

```
try
{
  NodeTypeProperty nodeTypeProperty =
api.CatalogNodeTypePropertyGet(28000000022144);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get node type property: {0}", info.Detail.Details)
  );
}
```

4.66.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.67 CatalogNodeTypePropertyGetByName Method

This method returns a node type property providing its name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.67.1 Syntax

C#

```
NodeTypeProperty CatalogNodeTypePropertyGetByName(
  string name,
  long nodeTypeId
)
```

4.67.1.1 Parameters

name
Type: System.String
Node type property name.

nodeTypeId
Type: System.Int64
Id of a node type the property belongs to.

4.67.1.2 Return Value

Type: NodeTypeProperty Returns the node type property object.

4.67.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.67.3 Examples

The following example shows how to get a node type property using the CatalogNodeTypePropertyGetByName method.

```
try
{
  NodeTypeProperty nodeTypeProperty =
api.CatalogNodeTypePropertyGetByName(  "Density", 26000000003382);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
       "Cannot get node type property: {0}", info.Detail.Details)
  );
}
```

4.68 CatalogNodeTypePropertyUpdate Method

This method updates the node type property in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.68.1 Syntax

C#

```
void CatalogNodeTypePropertyUpdate(
  NodeTypeProperty nodeTypeProperty
)
```

4.68.1.1 Parameters

nodeTypeProperty
Type: NetTerrain.WebApi.NodeTypeProperty
Node type property object with updated data.

4.68.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.68.3 Remarks

To update the node type property get the node type property object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

The node type field cannot be altered if the NotEditable parameter is true.

The node type id cannot be changed.

When altering the Name parameter (renaming the node type property) the new name must be unique. Only non-system properties can be renamed.

The system status of a property cannot be changed. The following parameters of system properties cannot be changed: Name, IsInProperties, Mandatory, LockList, Position.

The IsTypeField parameter cannot be changed. For type fields the following parameters cannot be changed: DefaultValue, NotEditable, IsUniqueForThisType, IsUniqueForAllTypes.

The LockList parameter cannot be set to 'true' if a property does not have any list values.

4.68.4 Examples

The following example shows how to update the node type property using the CatalogNodeTypePropertyUpdate method.

```
try
{
  // Get the node type property object.
  NodeTypeProperty nodeTypeProperty =
api.CatalogNodeTypePropertyGet(28000000022144);

  // Update node type parameters.
  nodeTypeProperty.Name = "Charm factor";
  nodeTypeProperty.Angle = 20;
  nodeTypeProperty.Bold = true;
  nodeTypeProperty.DefaultValue = "Default";
  nodeTypeProperty.Displayed = true;
  nodeTypeProperty.FillColor = "#123123";
  nodeTypeProperty.FontColor = "#321321";
  nodeTypeProperty.FontFamily = FontFamilies.LucidaConsole;
  nodeTypeProperty.FontSize = 12;
  nodeTypeProperty.IsInProperties = true;
  nodeTypeProperty.IsUniqueForAllTypes = false;
  nodeTypeProperty.IsUniqueForThisType = false;
  nodeTypeProperty.Italic = false;
  nodeTypeProperty.Justification = TextJustification.Center;
  nodeTypeProperty.LockList = false;
  nodeTypeProperty.OffsetX = 0;
  nodeTypeProperty.OffsetY = 0;
  nodeTypeProperty.Position = 3;
  nodeTypeProperty.TextAlign = TextAligns.Center;
  nodeTypeProperty.Underline = false;

  // Update the node type.
```

```
    api.CatalogNodeTypePropertyUpdate(nodeTypeProperty);
  }
  catch (FaultException<FaultInfo> info)
  {
    Console.WriteLine(string.Format(
        "Cannot update the node type: {0}", info.Detail.Details)
    );
  }
```

4.68.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.69 CatalogNodeTypeUpdate Method

This method updates the node type in the catalog.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.69.1 Syntax

C#

```
void CatalogNodeTypeUpdate(
  NodeType nodeType
)
```

4.69.1.1 Parameters

nodeType
Type: NetTerrain.WebApi.NodeType
Node type object with updated data.

4.69.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.69.3 Remarks

To update the node type get the node type object, change one or more necessary parameters and then run the update method for this object. The Id parameter should not be changed.

System node types cannot be updated. The IsSystem parameter cannot be altered either.

The type group of the node type cannot be changed.

When altering the Name parameter (renaming the node type) the new name must be unique.

When changing the ImageFileName or Backgroung parameters the image file corresponding to the new file name should already exist on the server. For uploading new image files the CatalogNodeTypeUploadImage(FileUploadAttributes) or CatalogNodeTypeUploadBackground(FileUploadAttributes) methods should be used. Use an empty string to remove the node icon or background image.

A favorite node type or a node type with existing node instances cannot be disabled.

The Background, VendorId, Width and Height parameters can only be set for node types that were modelled as Racks, Devices and Cards.

The HBLabelId (Hierarchy browser label) only accepts the property Id of properties belonging to the current node type.

The new TemplateId should correspond to a template node.

The CategoryId should correspond to a node category of the same type group.

The Display Rack Lines can only be set to true for the Rack type group.

4.69.4 Examples

The following example shows how to update the node type using CatalogNodeTypeUpdate method.

```
try
{
  // Get node type object.
  NodeType nodeType = api.CatalogNodeTypeGetByName("Antenna");

  // Update node type parameters.
  nodeType.Name = "Antennas";
  nodeType.DefaultHeight = 0.23f;
  nodeType.DefaultWidth = 0.45f;
  nodeType.Description = "Arrays et. el..";
  nodeType.IsEnabled = true;
  nodeType.IsFavorite = true;
  nodeType.KeepAspectRatio = false;
  nodeType.HBLabelId = 28000000022113;
  nodeType.DisplayRackLines = false;
  nodeType.DoubleClickBehavior = DoubleClickBehaviors.OpenURL;
  nodeType.DoubleClickBehaviorAttribute = "http://
www.katzenjammer.no";
  nodeType.ImageFileName = "BigAntenna.png";
  nodeType.Background = "Wilde.png";
  nodeType.CategoryId = 31000000000051;
  nodeType.TemplateId = 24000000000231;
  nodeType.VendorId = 8;
  nodeType.Width = 19f;
  nodeType.Height = 1.75f;

  // Update the node type.
  api.CatalogNodeTypeUpdate(nodeType);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot update the node type: {0}", info.Detail.Details)
  );
}
```

4.69.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.70 CatalogNodeTypeUploadBackground Method

This method uploads a new background for an existing node type.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.70.1 Syntax

C#

```
void CatalogNodeTypeUploadBackground(
  FileUploadAttributes fileUploadAttributes
)
```

4.70.1.1 Parameters

fileUploadAttributes
Type: NetTerrain.WebApi.FileUploadAttributes
Attributes of a new node type background.

4.70.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.70.3 Remarks

Images can only be uploaded for non-system node types belonging to Device, Rack or Card node type groups.

4.70.4 Examples

The following example shows how to upload an image using the CatalogNodeTypeUploadBackground method.

```
try
{
  // Get node type.
  NodeType deviceTypeToUpdate =  api.CatalogNodeTypeGetByName("ATM
Switches");
  FileStream stream = new FileStream(
    @"c:\ATM.png", FileMode.Open, FileAccess.Read);
    api.CatalogNodeTypeUploadBackground(new FileUploadAttributes()
{
      FileName = "ATM.png",
      FileByteStream = stream,  ObjectId = deviceTypeToUpdate.Id
    });

  // Release the stream.
  stream.Dispose();
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot upload node type background: {0}", info.Detail.Details)
  );
}
```

4.70.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.71 CatalogNodeTypeUploadImage Method

This method uploads a new image for an existing node type.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.71.1 Syntax

C#

```
void CatalogNodeTypeUploadImage(
  FileUploadAttributes fileUploadAttributes
)
```

4.71.1.1 Parameters

fileUploadAttributes
Type: NetTerrain.WebApi.FileUploadAttributes
Attributes of the new node type image.

4.71.2 Exceptions

Exception	**Condition**
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.71.3 Remarks

An image can only be uploaded for non-system node types.

4.71.4 Examples

The following example shows how to upload an image using the CatalogNodeTypeUploadImage method.

```
try
{
  // Get node type.
  NodeType nodeTypeToUpdate =  api.CatalogNodeTypeGetByName("PDUs");
  FileStream stream = new FileStream(
    @"c:\PDU.png", FileMode.Open, FileAccess.Read);
    api.CatalogNodeTypeUploadImage(new FileUploadAttributes()  {
      FileName = "PDU.png",
      FileByteStream = stream,  ObjectId = nodeTypeToUpdate.Id  });
  // Release the stream.
  stream.Dispose();
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot upload node type image: {0}", info.Detail.Details)
  );
}
```

4.71.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.72 CatalogVendorAdd Method

This method adds a new vendor.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.72.1 Syntax

C#

```
long CatalogVendorAdd(
  string vendorName
)
```

4.72.1.1 Parameters

vendorName
Type: System.String
Vendor name. This name must be unique throughout the vendors catalog.

4.72.1.2 Return Value

Type: **Int64** Returns the id of the newly created vendor.

4.72.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.72.3 Examples

The following example shows how to add a new vendor using the CatalogVendorAdd method.

```
try
{
  long vendorId = api.CatalogVendorAdd("Acme");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot add vendor: {0}", info.Detail.Details)
  );
}
```

4.72.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.73 CatalogVendorDelete Method

This method deletes a vendor.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.73.1 Syntax

C#

```
void CatalogVendorDelete(
  long vendorId
)
```

4.73.1.1 Parameters

vendorId
Type: System.Int64
The id of the vendor to delete.

4.73.2 Exceptions

Exception	**Condition**
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.73.3 Remarks

Vendors in use cannot be deleted.

4.73.4 Examples

The following example shows how to delete a vendor using the CatalogVendorDelete method.

```
try
{
  // Get existing vendor.
  Vendor vendor = api.CatalogVendorGetByName("Acme");

  // Delete the vendor.
  api.CatalogVendorDelete(vendor.Id);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot delete vendor: {0}", info.Detail.Details)
  );
}
```

4.73.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.74 CatalogVendorGet Method

This method gets a vendor object.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.74.1 Syntax

C#

```
Vendor CatalogVendorGet(
  long vendorId
)
```

4.74.1.1 Parameters

vendorId
Type: System.Int64
Vendor id.

4.74.1.2 Return Value

Type: Vendor Returns the vendor object.

4.74.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.74.3 Remarks

Unlike other objects, vendor ids start from 1.

4.74.4 Examples

The following example shows how to get a vendor using the CatalogVendorGet method.

```
try
{
  Vendor vendor = api.CatalogVendorGet(3);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get vendor: {0}", info.Detail.Details)
  );
}
```

4.74.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.75 CatalogVendorGetByName Method

This method gets a vendor object by name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.75.1 Syntax

C#

```
Vendor CatalogVendorGetByName(
  string vendorName
)
```

4.75.1.1 Parameters

vendorName
Type: System.String
Vendor name.

4.75.1.2 Return Value

Type: Vendor Returns the vendor object.

4.75.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.75.3 Examples

The following example shows how to get a vendor object using the CatalogVendorGetByName method.

```
try
{
  Vendor vendor = api.CatalogVendorGetByName("Acme");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get vendor: {0}", info.Detail.Details)
  );
}
```

4.75.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.76 CatalogVendorUpdate Method

This method updates a vendor.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.76.1 Syntax

C#

```
void CatalogVendorUpdate(
  Vendor vendor
)
```

4.76.1.1 Parameters

vendor
Type: NetTerrain.WebApi.Vendor
Vendor object with updated data.

4.76.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.76.3 Remarks

To update a vendor object, first retrieve the object, then change one or more parameters and then run the update method. The Id parameter should not be changed. Currently vendors only have one property, which is the name.

When altering the Name parameter (renaming the vendor) the new name must be unique in the catalog of vendors.

4.76.4 Examples

The following example shows how to update the vendor using the CatalogVendorUpdate method.

```
try
{
  // Get vendor object.
  Vendor vendor = api.CatalogVendorGetByName("Acme");

  // Update vendor parameters.
  vendor.Name = "BeepBeep";

  // Update the vendor.
  api.CatalogVendorUpdate(vendor);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot update the vendor: {0}", info.Detail.Details)
  );
}
```

4.76.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.77 DiagramGetHeight Method

This method gets the height of a diagram.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.77.1 Syntax

C#

```
float DiagramGetHeight(
  long diagramId
)
```

4.77.1.1 Parameters

diagramId
Type: System.Int64
Diagram id.

4.77.1.2 Return Value

Type: **Single** Returns the diagram height values in ‘points’ (one point equals 0.01 inches).

4.77.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.77.3 Examples

The following example shows how to retrieve the height of the specified diagram id using the DiagramGetHeight method.

```
try
{
  float height = api.DiagramGetHeight(24000000018239);
  Console.WriteLine("Value = " + height);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
// The example could return something like this:
// Value = 850
```

4.77.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.78 DiagramGetLinksByTypeId Method

This method gets a list of link names providing the type and the diagram.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.78.1 Syntax

C#

```
Dictionary<long, string> DiagramGetLinksByTypeId(
  long diagramId,
  long linkTypeId,
  bool includeSubDiagrams
)
```

4.78.1.1 Parameters

diagramId
Type: System.Int64
Diagram id.

linkTypeId
Type: System.Int64
Link type id.

includeSubDiagrams
Type: System.Boolean
If set to true links are also searched in any child diagrams.

4.78.1.2 Return Value

Type: **Dictionary**(**Int64**, **String**) Returns a dictionary containing 'link id' - 'link name' pairs or an empty dictionary if no links were found.

4.78.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.78.3 Examples

The following example shows how to retrieve all the link names of the specified diagram id and type id using the DiagramGetLinksWithNamesByTypeId method.

```
try
{
  Dictionary<long, string> list = api.DiagramGetLinksByTypeId(
24000000018239, 27000000000033, false);
  foreach (KeyValuePair<long, string> entry in list)  {
    Console.WriteLine(string.Format(
      "Property id: {0}, name: {1}", entry.Key, entry.Value));
  }
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get a list of links: {0}", info.Detail.Details)
    );
}
// The example retrieves an output such as the following:
// Link id: 25000000006219, name: Link1
// Link id: 25000000006220, name: Link2
// Link id: 25000000006223, name: Ali G
// Link id: 25000000006224, name: Me Julie
// Link id: 25000000007902, name: BooyahShakkah
```

4.78.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.79 DiagramGetMarginSize Method

This method gets the margin size of a diagram.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.79.1 Syntax

C#

```
float DiagramGetMarginSize(
  long diagramId
)
```

4.79.1.1 Parameters

diagramId
Type: System.Int64
Diagram id.

4.79.1.2 Return Value

Type: **Single** Returns the diagram margin size value in ‘points’ (one point equals 0.01 inches).

4.79.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.79.3 Examples

The following example shows how to get the margin size of the specified diagram using the DiagramGetMarginSize method.

```
try
{
  float margin = api.DiagramGetMarginSize(24000000018239);
  Console.WriteLine("Value = " + margin);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
// Output:
// Value = 50
```

4.79.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.80 DiagramGetNodesByTypeGroup Method

This method gets a dictionary of node ids and names providing the type group and the diagram.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.80.1 Syntax

C#

```
Dictionary<long, string> DiagramGetNodesByTypeGroup(
  long diagramId,
  NodeTypeGroups typeGroup,
  HierarchySearchModes searchMode,
    int searchDepth
)
```

4.80.1.1 Parameters

diagramId
Type: System.Int64
Diagram id.

typeGroup
Type: NetTerrain.WebApi.NodeTypeGroups
Node type group value of NodeTypeGroups enumeration.

searchMode
Type: NetTerrain.WebApi.HierarchySearchModes
Nodes search mode value of HierarchySearchModes enumeration.

searchDepth
Type: System.Int32
Number of hierarchy levels for nodes searching if searchMode is set to N_Levels.

4.80.1.2 Return Value

Type: **Dictionary**(**Int64**, **String**) Returns a dictionary containing 'node id' - 'node name' pairs or an empty dictionary if no nodes were found.

4.80.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.80.3 Examples

The following example shows how to retrieve all the nodes of the specified diagram id and type id using the DiagramGetNodesByTypeGroup method.

```
try
{
  Dictionary<long, string> list = api.DiagramGetNodesByTypeGroup(
24000000018239, NodeTypeGroups.Device,
HierarchySearchDepth.N_Levels, 3);
  foreach (KeyValuePair<long, string> entry in list)  {
    Console.WriteLine(string.Format("Device id: {0}, name: {1}",
entry.Key, entry.Value));
  }
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Cannot get a list of nodes: {0}",
info.Detail.Details)
    );
}
// The example retrieves something like the following foo:
// Device id: 24000000018447, name: Switch 1
// Device id: 24000000018454, name: Switch 2
```

4.80.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.81 DiagramGetNodesByTypeId Method

Gets a list of nodes of a selected type on a selected diagram.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.81.1 Syntax

C#

```
Dictionary<long, string> DiagramGetNodesByTypeId(
  long diagramId,
  long nodeTypeId,
  HierarchySearchModes searchMode,
  int searchDepth
)
```

4.81.1.1 Parameters

diagramId
Type: System.Int64
Diagram id.

nodeTypeId
Type: System.Int64
Node type id.

searchMode

Type: NetTerrain.WebApi.HierarchySearchModes

Nodes search mode value of HierarchySearchModes enumeration.

searchDepth

Type: System.Int32

Number of hierarchy levels for nodes searching if searchMode is set to N_Levels.

4.81.1.2 Return Value

Type: **Dictionary**(**Int64**, **String**) Returns a dictionary containing 'node id' - 'node name' pairs or an empty dictionary if no nodes were found.

4.81.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.81.3 Examples

The following example shows how to retrieve all the nodes of the specified diagram id and type id using the DiagramGetNodesByTypeId method.

```
try
{
  Dictionary<long, string> list = api.DiagramGetNodesByTypeId(
24000000018239, 27000000000033, HierarchySearchDepth.N_Levels, 2);
  foreach (KeyValuePair<long, string> entry in list)  {
    Console.WriteLine(string.Format("Node id: {0}, name: {1}",
entry.Key, entry.Value));  }
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Cannot get a list of nodes: {0}",
info.Detail.Details)
    );
}
// The example retrieves something like the following foo:
// Node id: 24000000018443, name: Workstation 1
// Node id: 24000000018444, name: Workstation 2
```

4.81.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.82 DiagramGetWidth Method

This method gets the width of a diagram.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.82.1 Syntax

C#

```
float DiagramGetWidth(
  long diagramId
)
```

4.82.1.1 Parameters

diagramId
Type: System.Int64
Diagram id.

4.82.1.2 Return Value

Type: **Single** Returns the diagram width value in ‘points’ (one point equals 0.01 inches).

4.82.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.82.3 Examples

The following example shows how to retrieve the width of the specified diagram id using the DiagramGetWidth method.

```
try
{
  float width = api.DiagramGetWidth(24000000018239);
  Console.WriteLine("Value = " + width);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.82.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.83 InstanceMoveToFront Method

This method moves an instance on a diagram to the front.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.83.1 Syntax

C#

```
void InstanceMoveToFront(
  long diagramId,
  long instanceId
)
```

4.83.1.1 Parameters

diagramId
Type: System.Int64
Diagram id.

instanceId
Type: System.Int64
Instance Id. Should correspond to a Node, Link or FreeText id.

4.83.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.83.3 Remarks

In netTerrain terms, an instance is any selectable object rendered on a diagram, such as a node, a link or a free text. This method will move an instance to the front if it is a Node, a Link or a FreeText object and it will calculate the so-called zOrder of that instance as the maximum existing zOrder of any other objects on that diagram, plus 1.

4.83.4 Examples

The following example shows how to move the specified instance to the front of the diagram (on top of other objects) using the InstanceMoveToFront method.

```
try
{
  api.InstanceMoveToFront(24000000014087, 24000000031942);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.84 InstanceSendToBack Method

This method sends an instance on a diagram to the back.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.84.1 Syntax

C#

```
void InstanceSendToBack(
  long diagramId,
  long instanceId
)
```

4.84.1.1 Parameters

diagramId
Type: System.Int64
Diagram id.

instanceId
Type: System.Int64
Instance Id. Should correspond to a Node, Link or FreeText id.

4.84.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.84.3 Remarks

In netTerrain terms, an instance is any selectable object rendered on a diagram, such as a node, a link or a free text. This method will send an instance to the back if it is a Node, a Link or a FreeText object and it will calculate the so-called zOrder of that instance as the minimum existing zOrder of any other objects on that diagram, minus 1.

4.84.4 Examples

The following example shows how to send the specified instance to the back of the diagram (behind other objects) using the InstanceSendToBack method.

```
try
{
  api.InstanceSendToBack(24000000014087, 24000000031942);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.85 LinkDelete Method

This method deletes a link.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.85.1 Syntax

C#

```
void LinkDelete(
  long linkId
)
```

4.85.1.1 Parameters

linkId
Type: System.Int64
Link id.

4.85.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.85.3 Examples

The following example shows how to delete the specified link using the LinkDelete method.

```
try
{
  api.LinkDelete(25000000019835);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.85.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.86 LinkGetPropertyValue Method

This method gets the value of a link property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.86.1 Syntax

C#

```
string LinkGetPropertyValue(
  long linkId,
  long linkPropertyId
)
```

4.86.1.1 Parameters

linkId
Type: System.Int64
Link id.

linkPropertyId
Type: System.Int64
Link property id in the catalog.

4.86.1.2 Return Value

Type: **String** Returns the value of the specified property.

4.86.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.86.3 Examples

The following example shows how to get the value of the specified property and link using the LinkGetPropertyValue method.

```
try
{
  string value = api.LinkGetPropertyValue(25000000019835,
29000000000555);
  Console.WriteLine("Value = " + value);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.87 LinkGetPropertyValueByName Method

This method gets the value of a link property passing the property name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.87.1 Syntax

C#

```
string LinkGetPropertyValueByName(
  long linkId,
  string linkPropertyName
)
```

4.87.1.1 Parameters

linkId
Type: System.Int64
Link id.

linkPropertyName
Type: System.String
Link property name in the catalog.

4.87.1.2 Return Value

Type: **String** Returns the value of the specified property.

4.87.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.87.3 Examples

The following example shows how to get the value of the specified property (by name) and link using the LinkGetPropertyValueByName method.

```
try
{
  string value = api.LinkGetPropertyValueByName(25000000019835, "A
port");
  Console.WriteLine("Value = " + value);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.88 LinkGetTypeId Method

This method gets the type id of a link.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.88.1 Syntax

C#

```
long LinkGetTypeId(
  long linkId
)
```

4.88.1.1 Parameters

linkId
Type: System.Int64
Link id.

4.88.1.2 Return Value

Type: **Int64** Returns the link type id of a specified link.

4.88.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.88.3 Examples

The following example shows how to get the type id of the specified link using the LinkGetTypeId method.

```
try
{
  long typeId = api.LinkGetTypeId(25000000019834);
  Console.WriteLine("Type id = " + typeId);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.88.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.89 LinkInsert Method

This method inserts a link with a specific name and of a specific type providing the end nodes.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.89.1 Syntax

C#

```
long LinkInsert(
  string name,
  long typeId,
  long node1Id,
  long node2Id
)
```

4.89.1.1 Parameters

name
Type: System.String
Name of the new link.

typeId
Type: System.Int64
Type id of the new link.

node1Id
Type: System.Int64
Id of the starting node.

node2Id
Type: System.Int64
Id of the ending node.

4.89.1.2 Return Value

Type: **Int64** Returns the id of the inserted link.

4.89.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.89.3 Remarks

In netTerrain, the endpoints of a link can be on two different diagrams. In this case an inter diagram link will be created automatically.

4.89.4 Examples

The following example shows how to insert a link using the LinkInsert method.

```
try
{
  long linkId = api.LinkInsert(  "My New Link", 27000000000124,
24000000031830, 24000000031827);
  Console.WriteLine("LinkId = " + linkId);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.89.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.90 LinkPropertyUpdate Method

This method updates a link property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.90.1 Syntax

C#

```
void LinkPropertyUpdate(
  long linkId,
  long propertyId,
  string value
)
```

4.90.1.1 Parameters

linkId
Type: System.Int64
Id of the link that needs to be updated.

propertyId
Type: System.Int64 Id of the property to be updated.

value
Type: System.String

4.90.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.90.3 Examples

The following example shows how to update the value of the specified property and link using the LinkPropertyUpdate method.

```
try
{
  // 25000000020940 - Link Id - instance of type "Pet".
  // 29000000000555 - "Sound" property id.
  api.LinkPropertyUpdate(25000000020940, 29000000000555, "Gobble
gobble");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.91 LinkTypeGetId Method

This method gets the id of the link type providing its name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.91.1 Syntax

C#

```
long LinkTypeGetId(
  string linkTypeName
)
```

4.91.1.1 Parameters

linkTypeName
Type: System.String
Link type name.

4.91.1.2 Return Value

Type: **Int64** Returns the link type id.

4.91.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.91.3 Examples

The following example shows how to get the Id of the specified link type (by name) using the LinkTypeGetId method.

```
try
{
  long linkTypeId = api.LinkTypeGetId("Big huge circuit");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format(
      "Cannot get link type id: {0}", info.Detail.Details)
  );
}
```

4.91.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.92 NodeDelete Method

This method deletes a node.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.92.1 Syntax

C#

```
void NodeDelete(
  long nodeId
)
```

4.92.1.1 Parameters

nodeId
Type: System.Int64
Node id.

4.92.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.92.3 Examples

The following example shows how to delete the specified node using the NodeDelete method.

```
try
{
  api.NodeDelete(24000000031827);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.92.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.93 NodeGetPropertyValue Method

This method gets the value of a node property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.93.1 Syntax

C#

```
string NodeGetPropertyValue(
  long nodeId,
  long nodePropertyId
)
```

4.93.1.1 Parameters

nodeId
Type: System.Int64
Node id.

nodePropertyId
Type: System.Int64
Node property id in the catalog.

4.93.1.2 Return Value

Type: **String** Returns the value of the specified property.

4.93.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.93.3 Examples

The following example shows how to get the value of the specified property and node using the NodeGetPropertyValue method.

```
try
{
  string value = api.NodeGetPropertyValue(24000000029203,
28000000019353);
  Console.WriteLine("Value = " + value);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.94 NodeGetPropertyValueByName Method

This method gets the value of a node property by passing the property name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.94.1 Syntax

C#

```
string NodeGetPropertyValueByName(
  long nodeId,
  string nodePropertyName
)
```

4.94.1.1 Parameters

nodeId
Type: System.Int64
Node id.

nodePropertyName
Type: System.String
Node property name (as defined in the catalog).

4.94.1.2 Return Value

Type: **String** Returns the value of the selected property.

4.94.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.94.3 Examples

The following example shows how to get the value of the specified property (by name) and node using the NodeGetPropertyValueByName method.

```
try
{
  string value = api.NodeGetPropertyValueByName(24000000029203,
"Depth");
  Console.WriteLine("Value = " + value);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.95 NodeGetTypeGroup Method

This method gets the type group of a node.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.95.1 Syntax

C#

```
NodeTypeGroups NodeGetTypeGroup(
  long nodeId
)
```

4.95.1.1 Parameters

nodeId
Type: System.Int64
Node id.

4.95.1.2 Return Value

Type: NodeTypeGroups Returns the enumeration value of the NodeTypeGroups enumeration that corresponds to the specified node.

4.95.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.95.3 Examples

The following example shows how to get the type group of specified node using the NodeGetTypeGroup method.

```
try
{
  NodeTypeGroups typeGroup = api.NodeGetTypeGroup(24000000029203);
  Console.WriteLine("Node type group = " + typeGroup);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.95.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.96 NodeGetTypeId Method

This method gets the type id of a node.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.96.1 Syntax

C#

```
long NodeGetTypeId(
  long nodeId
)
```

4.96.1.1 Parameters

nodeId
Type: System.Int64
Node id.

4.96.1.2 Return Value

Type: **Int64** Returns the id of the node type that corresponds to the specified node.

4.96.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.96.3 Examples

The following example shows how to get the TypeId of specified node using the NodeGetTypeId method.

```
try
{
  long typeId = api.NodeGetTypeId(24000000029203);
  Console.WriteLine("Type id = " + typeId);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.96.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.97 NodeInsert Method

This method inserts a node with a specific name and of a specific type providing the parent diagram.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.97.1 Syntax

C#

```
long NodeInsert(
  string name,
  long parentId,
  long typeId
)
```

4.97.1.1 Parameters

name
Type: System.String
Name of the inserted node.

parentId
Type: System.Int64
Id of the parent diagram where we will insert the node.

typeId
Type: System.Int64
Type id of the inserted node.

4.97.1.2 Return Value

Type: **Int64** Returns the id of the inserted node.

4.97.2 Exceptions

Exception	**Condition**
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.97.3 Examples

The following example shows how to insert a node on a diagram using the NodeInsert method.

```
try
{
  long nodeId = api.NodeInsert(  "DataCenter2", 24000000014087,
26000000003700);
  Console.WriteLine("NodeId = " + nodeId);
}
catch (FaultException<FaultInfo> info)
```

```
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.97.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.98 NodePropertyUpdate Method

This method updates a node property.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.98.1 Syntax

C#

```
void NodePropertyUpdate(
  long nodeId,
  long propertyId,
  string value
)
```

4.98.1.1 Parameters

nodeId
Type: System.Int64
Id of the node to be updated.

propertyId
Type: System.Int64
Id of property to be updated.

value
Type: System.String
New property value.

4.98.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.98.3 Examples

The following example shows how to update the value of a given property and node using the NodePropertyUpdate method.

```
try
{
  // 24000000030435 - Node Id - instance of type "Animal”.
  // 28000000034803 - Id of the property "Kind".
  api.NodePropertyUpdate(24000000030435, 28000000034803,
"Cucaracha");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.99 NodeReparent Method

This method moves a node to another diagram.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.99.1 Syntax

C#

```
void NodeReparent(
  long nodeId,
  long newParentId
)
```

4.99.1.1 Parameters

nodeId
Type: System.Int64
Id of the node to be moved.

newParentId
Type: System.Int64
The id of the new parent diagram.

4.99.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.99.3 Examples

The following example shows how to reparent a node (i.e. change its ancestry) by using the NodeReparent method.

```
try
{
  api.NodeReparent(24000000014087, 24000000012754);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.99.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.100 NodeSetCanMove Method

This method updates the 'CanMove' setting of a visNode based on the node id and diagram id.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.100.1 Syntax

C#

```
void NodeSetCanMove(
  long diagramId,
  long nodeId,
  bool canMove
)
```

4.100.1.1 Parameters

diagramId
Type: System.Int64
Id of the diagram containing the node.

nodeId
Type: System.Int64
Id of the node.

canMove
Type: System.Boolean
New value of the 'CanMove' setting.

4.100.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.100.3 Remarks

The 'CanMove' flag is one of several parameters associated with a visNode.

4.100.4 Examples

The following example shows how override the CanMove flag for a specified node using the NodeSetCanMove method.

```
try
{
  api.NodeSetCanMove(24000000014087, 24000000030435, false);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.101 NodeSetHeight Method

This method sets the height of a visNode based on the node id and diagram id.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.101.1 Syntax

C#

```
void NodeSetHeight(
  long diagramId,
  long nodeId,
  float height
)
```

4.101.1.1 Parameters

diagramId
Type: System.Int64
Id of the diagram containing the node.

nodeId
Type: System.Int64
Id of the node.

height
Type: System.Single
New height value in 'points' (one point equals 0.01 inches).

4.101.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.101.3 Remarks

The 'Height' parameter is one of several parameters associated with a visNode.

4.101.4 Examples

The following example shows how override the CanMove flag for a specified node using the NodeSetCanMove method.

```
try
{
  // 120 diagram points equals 1.2 inches.
  api.NodeSetHeight(24000000014087, 24000000030435, 120);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.102 NodeSetWidth Method

This method sets the width of a visNode based on the node id and diagram id.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.102.1 Syntax

C#

```
void NodeSetWidth(
  long diagramId,
  long nodeId,
  float width
)
```

4.102.1.1 Parameters

diagramId
Type: System.Int64
Id of the diagram containing the node.

nodeId
Type: System.Int64
Id of the node.

width
Type: System.Single
New width value in 'points' (one point equals 0.01 inches).

4.102.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.102.3 Remarks

The 'Width' parameter is one of several parameters associated with a visNode.

4.102.4 Examples

```
try
{
  // 80 diagram points equals 0.8 inches.
  api.NodeSetWidth(24000000014087, 24000000030435, 80);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.103 NodeSetX Method

This method sets the X coordinate of the top-left corner of a visNode based on the node id and diagram id.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.103.1 Syntax

C#

```
void NodeSetX(
  long diagramId,
  long nodeId,
  float x
)
```

4.103.1.1 Parameters

diagramId
Type: System.Int64
Id of the diagram containing the node.

nodeId
Type: System.Int64
Id of the node.

x
Type: System.Single
New X coordinate in 'points' (one point equals 0.01 inches).

4.103.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.103.3 Remarks

The 'X' coordinate is one of several parameters associated with a visNode.

4.103.4 Examples

The following example shows how to override the X coordinate of a node by using the NodeSetX method.

```
try
{
  // Set the node 0.2 inches to the right of the diagram left margin.
  api.NodeSetX(24000000014087, 24000000030435, 20);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.103.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.104 NodeSetY Method

This method sets the Y coordinate of the top-left corner of a visNode based on the node id and diagram id.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.104.1 Syntax

C#

```
void NodeSetY(
  long diagramId,
  long nodeId,
  float y
)
```

4.104.1.1 Parameters

diagramId
Type: System.Int64
Id of the diagram containing the node.

nodeId
Type: System.Int64
Id of the node.

y
Type: System.Single
New Y coordinate in 'points' (one point equals 0.01 inches).

4.104.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.104.3 Remarks

The 'Y' coordinate is one of several parameters associated with a visNode.

4.104.4 Examples

The following example shows how to override the Y coordinate of a node by using the NodeSetY method.

```
try
{
  // Set the node 0.2 inches below the diagram's top margin.
  api.NodeSetY(24000000014087, 24000000030435, 20);
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.104.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.105 NodesGetByName Method

Gets a list of nodes with the specified name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.105.1 Syntax

C#

```
List<long> NodesGetByName(
  string nodeName
)
```

4.105.1.1 Parameters

nodeName
Type: System.String
Node name.

4.105.1.2 Return Value

Type: **List**(**Int64**)

4.105.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.105.3 Remarks

The result is the list of node ids as a node name is not unique.

4.105.4 Examples

The following example shows how to get the list of node Ids using the NodesGetByName method.

```
try
{
  long[] nodeIds = api.NodesGetByName("My Node");
}
catch (FaultException<FaultInfo> info)
```

```
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.105.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.106 NodeTypeGetId Method

This method gets the id of the node type providing its name.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.106.1 Syntax

C#

```
long NodeTypeGetId(
  string nodeTypeName
)
```

4.106.1.1 Parameters

nodeTypeName
Type: System.String
Node type name.

4.106.1.2 Return Value

Type: **Int64** Returns the node type id.

4.106.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.106.3 Examples

The following example shows how to get the Id of the specified node type (by name) using the NodeTypeGetId method.

```
try
{
  long nodeTypeId = api.NodeTypeGetId("My Node Type");
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Cannot get node type id: {0}",
info.Detail.Details));
}
```

4.106.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.107 TestConnection Method

This method tests the opened channel between the server and the client. This method does nothing unless a problem is detected, in which case it throws a FaultException(TDetail) error, including the problem description.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.107.1 Syntax

C#

```
void TestConnection()
```

4.107.2 Exceptions

Exception	Condition
[!:FaultException]	Raised in case of a fault on method execution.

4.107.3 Examples

This example shows how to check the channel connection created between a client application and the netTerrain instance.

```
// Create connection.
ChannelFactory<WebApiReference.INetTerrainWebApi factory =
CreateConnectionFactory(
  @"https://myserver.com/netTerrain/WebApi/NetTerrainWebApi.svc",
"webApiUser",  "webApiUserPassword");
WebApiReference.INetTerrainWebApi api = factory.CreateChannel();
try
{
  api.TestConnection();
}
catch (Exception ex)
{
  Console.WriteLine(string.Format("Connection failed: {0}",
ex.Message));
}
```

4.107.4 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.108 VisNodeSetAttribute Method

This method updates a set of attributes of a visNode based on the node id and diagram id.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.108.1 Syntax

C#

```
void VisNodeSetAttribute(
  long diagramId,
  long nodeId,
  Dictionary<string, string> attributes
)
```

4.108.1.1 Parameters

diagramId
Type: System.Int64
Id of the diagram containing the node.

nodeId
Type: System.Int64
Id of the node.

attributes
Type: System.Collections.Generic.Dictionary(**String**, **String**)
A dictionary of 'attribute name' - 'new attribute value' pairs. Possible attribute names include: X, Y, Height, Width, CanMove. Attribute names are not case sensitive.

4.108.2 Exceptions

Exception	Condition
FaultException(TDetail)	Raised in case of a fault on method execution. Stores details in FaultInfo object.

4.108.3 Remarks

Attributes belong to a visNode - a visible representation of a node on a particular diagram. A particular visNode instance could be determined by ids of its base node and diagram it is placed on - a single node could have reference nodes on different diagrams, each of them will require its own visNode.

4.108.4 Examples

The following example shows how to set the attributes of a VisNode using the VisNodeSetAttribute method.

```
try
{
  api.VisNodeSetAttribute(24000000014087, 24000000030435,
    new Dictionary<string, string> {
      { "x", "20" },
      { "y", "20" },
      { "width", "80" },
      { "height", "120" },
      { "canMove", "false" }
    }
  );
}
catch (FaultException<FaultInfo> info)
{
  Console.WriteLine(string.Format("Error: {0}",
info.Detail.Details));
}
```

4.108.5 See Also

INetTerrainWebApi Interface

NetTerrain.WebApi Namespace

4.109 InstanceEffects Enumeration

This enumeration type sets effects for an overriden instance.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.109.1 Syntax

C#

```
public enum InstanceEffects
```

4.109.2 Members

Member name	Value	Description
NoEffect	0	No Effect.
BlinkObject	1	Blink Object.
RectangleBlink	2	Rectangle Blink.
RedTriangleIndicator	3	Red Triangle Indicator.
OrangeTriangleIndicator	4	Orange Triangle Indicator.
GreenTriangleIndicator	5	Green Triangle Indicator.

4.109.3 See Also

NetTerrain.WebApi Namespace

4.110 LinkCategory Class

This class contains parameters of a link category.

4.110.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.LinkCategory

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.110.2 Syntax

C#

```
public class LinkCategory
```

The **LinkCategory** type exposes the following members.

4.110.3 Constructors

	Name	Description
	LinkCategory	

4.110.4 Fields

	Name	Description
	Id	Category id.
	ImageFileName	Image file name for a link category icon image.
	IsFavorite	If set to true makes the category a favorite.
	Name	Category name.
	ParentId	Parent category Id ('0' if there is no a parent category).

4.110.5 See Also

NetTerrain.WebApi Namespace

4.111 LinkOverride Class

This class contains parameters of a link property override.

4.111.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.LinkOverride

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.111.2 Syntax

C#

```
public class LinkOverride
```

The **LinkOverride** type exposes the following members.

4.111.3 Constructors

	Name	Description
	LinkOverride	

4.111.4 Fields

	Name	Description
	Color	Link's hex color code override (Example: '#000000').
	EndArrow	Override of an arrow style at link's end.
	Id	Link override id.
	IsOverride	When set to false does not set any override for this list value.
	LinkStyle	Link style override.
	ListValue	Override's list value.
	PropertyId	Corresponding link property id.
	Rule	Override rule.

	Name	Description
◆	StartArrow	Override of an arrow style at link's start.
◆	Thickness	Link's thickness override.

4.111.5 See Also

NetTerrain.WebApi Namespace

4.112 LinkStyles Enumeration

This enumeration type sets styles of lines used to display links.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.112.1 Syntax

C#

```
public enum LinkStyles
```

4.112.2 Members

Member name	Value	Description
Solid	0	Solid line.
Dash	1	Dashed line.
DashDot	2	"Dash-Dot" line.
DashDotDot	3	"Dash-Dot-Dot" line.
Dot	4	Dotted line.

4.112.3 See Also

NetTerrain.WebApi Namespace

4.113 LinkType Class

This class represents a link type object.

4.113.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.LinkType

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.113.2 Syntax

C#

```
public class LinkType
```

The **LinkType** type exposes the following members.

4.113.3 Constructors

	Name	Description
	LinkType	

4.113.4 Fields

	Name	Description
	CategoryId	Link type category id ('0' if there is no any associated category).
	Color	Link's hex color code (Example: '#000000').
	EndArrow	End arrow style.
	Id	Link type id.
	IsEnabled	If set to true enables the type.

	Name	Description
	IsFavorite	If set to true makes the type a favorite.
	IsMatchingPortConnectors	If set to true sets 'Matching Port Connectors' restriction on the type.
	IsSnappedToEdge	If set to true turns on 'Snapped To Edge' option.
	IsSystem	If set to true indicates that the link type belongs to system types and cannot be altered or deleted.
	LinkStyle	Link style.
	Name	Link type name.
	StartArrow	Start arrow style.
	Thickness	Link thickness.

4.114 LinkTypeProperty Class

This class contains parameters of a link type property.

4.114.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.LinkTypeProperty

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.114.2 Syntax

C#

```
public class LinkTypeProperty
```

The **LinkTypeProperty** type exposes the following members.

4.114.3 Constructors

	Name	Description
	LinkTypeProperty	

4.114.4 Fields

	Name	Description
	Angle	Angle for property's text displaying (0..360).
	Bold	If set to true makes displayed property text bold.
	DefaultValue	Default value of the property.
	Displayed	If set to true makes the property displayed on a diagram.
	FillColor	Text background hex color code for a displayed property (Example: '#000000').
	FontColor	Text font hex color code for a displayed property (Example: '#000000').
	FontFamily	Font family for a displayed property.
	FontSize	Text font size for a displayed property.
	Id	Property id.
	IsInProperties	If set to true the property is displayed in properties list when a link of this type is selected on a diagram.
	IsSystem	If set to true indicates that the property belongs to system properties and cannot be deleted.
	IsTypeField	If true indicates that this property is a reference to a host link type.
	IsUniqueForAllTypes	If set to true makes a property unique for all link types.
	IsUniqueForThisType	If set to true makes a property unique for this link type.
	Italic	If set to true makes displayed property text italic.

	Name	Description
◆	Justification	Text justification mode relatively to the host link for a displayed property.
◆	LinkTypeId	Link type id.
◆	LockList	If set to true locks the list of values if the property has it.
◆	Mandatory	If set to true makes the property mandatory.
◆	Name	Property name.
◆	OffsetX	Text offset X for a displayed property.
◆	OffsetY	Text offset Y for a displayed property.
◆	Position	Property position number in a list of properties of this link type.
◆	TextAlign	Text alignment for a displayed property.
◆	Underline	If set to true makes displayed property text underlined.

4.114.5 See Also

NetTerrain.WebApi Namespace

4.115 NewNodeTypeInfo Class

This class combines parameters of a new node type in conjunction with the CatalogAddNodeType(NewNodeTypeInfo) method - an older version of the node type creation method for versions compatibility. It is adviced to use the newer method CatalogNodeTypeAdd(String, NodeTypeGroups, String, Boolean).

4.115.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.NewNodeTypeInfo

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.115.2 Syntax

C#

```
public class NewNodeTypeInfo : IDisposable
```

The **NewNodeTypeInfo** type exposes the following members.

4.115.3 Constructors

	Name	Description
	NewNodeTypeInfo	

4.115.4 Methods

	Name	Description
	Dispose	Closes image byte stream if opened.

4.115.5 Fields

	Name	Description
	DefaultHeight	Default node height in inches.
	DefaultWidth	Default node width in inches.
	ImageByteStream	Byte stream for uploading the icon image file to the server.
	ImageFileName	Specifies the image file name for a node type icon image.
	IsEnabled	If set to true enables the type.
	IsFavourite	If set to true makes the type a favorite.
	Name	Node type name. Must be unique for all node types throughout the catalog.

	Name	Description
	TypeGroupNumber	The type group number based on values of the NodeTypeGroups enumeration. Possible type groups include: Node, Device, Rack, Card.

4.116 NodeCategory Class

This class contains parameters of a node category.

4.116.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.NodeCategory

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.116.2 Syntax

C#

```
public class NodeCategory
```

The **NodeCategory** type exposes the following members.

4.116.3 Constructors

	Name	Description
	NodeCategory	

4.116.4 Fields

	Name	Description
	Id	Category id.
	ImageFileName	Image file name for a node category icon image.
	IsFavorite	If set to true makes the category a favorite.

	Name	Description
🔷	Name	Category name.
🔷	ParentId	Parent category Id ('0' if there is no a parent category).
🔷	TypeGroup	The type group of the category.

4.116.5 See Also

NetTerrain.WebApi Namespace

4.117 NodeOverride Class

This class contains parameters of a node property override.

4.117.1 Inheritance Hierarchy

System.Object NetTerrain.WebApi.NodeOverride

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.117.2 Syntax

C#

```
public class NodeOverride
```

The **NodeOverride** type exposes the following members.

4.117.3 Constructors

	Name	Description
🔷	NodeOverride	

4.117.4 Fields

	Name	Description
◆	Id	Override id.
◆	ImageFileName	Overriding image file name.
◆	InstanceEffect	Instance effect if this list value is set as an override.
◆	IsOverride	When set to false does not set any override for this list value.
◆	ListValue	Override's list value.
◆	ParentEffect	Effect on instance's parent if this list value is set as an override.
◆	PropertyId	Corresponding node property id.
◆	Rule	Override rule.
◆	UpwardsPropagation	Upwards propagation of an override effect if this list value is set as an override.

4.117.5 See Also

NetTerrain.WebApi Namespace

4.118 NodeType Class

This class represents a node type object.

4.118.1 Inheritance Hierarchy

System.Object NetTerrain.WebApi.NodeType

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.118.2 Syntax

C#

```
public class NodeType
```

The **NodeType** type exposes the following members.

4.118.3 Constructors

	Name	Description
	NodeType	

4.118.4 Fields

	Name	Description
	Background	Background image for modelled objects as Racks, Devices or Cards.
	CategoryId	Node type category id ('0' if there is no any associated category).
	DefaultHeight	Default node height in inches.
	DefaultWidth	Default node width in inches.
	Description	Node type description.
	DisplayRackLines	If set to true displays rack lines on a rack diagram. Not used for other node type groups.
	DoubleClickBehavior	Double click behavior for nodes of this node type.
	DoubleClickBehaviorAttribute	Double click behavior attribute referenced by DoubleClickBehavior parameter: should contain diagram id for GoToDiagram option or URL for OpenURL option.
	HBLabelId	Hierarchy browser label. Sets an id of this node type's property to be used as a label in the project's hierarchy tree (hierarchy browser).

	Name	Description
◆	Height	Physical height in inches for objects modeled as Racks, Devices or Cards.
◆	Id	Node type id.
◆	ImageFileName	Image file name for a node type icon image.
◆	IsEnabled	If set to true enables the type.
◆	IsFavorite	If set to true makes the type a favorite.
◆	IsSystem	If set to true indicates that the node type belongs to a system type and cannot be altered or deleted.
◆	KeepAspectRatio	If set to true keeps a constant type image aspect ratio.
◆	Name	Node type name.
◆	TemplateId	Node type print template id ('0' if there is no associated template).
◆	TypeGroup	The type group based on the NodeTypeGroups enumeration.
◆	VendorId	Vendor id for modelled objects as Racks, Devices or Cards ('0' if there is no any associated vendor).
◆	Width	Physical width in inches for objects modeled as Racks, Devices or Cards.

4.118.5 See Also

NetTerrain.WebApi Namespace

4.119 NodeTypeGroups Enumeration

This enumeration type describes groups of node types used in netTerrain.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.119.1 Syntax

C#

```
public enum NodeTypeGroups
```

4.119.2 Members

Member name	Value	Description
Undefined	0	
Node	1	Generic nodes.
Document	2	Documents.
Comment	3	Comments.
Stamp	4	Stamps.
Shape	5	Shape nodes.
Picture	6	Free drawings.
Device	7	Devices.
Rack	8	Racks.
Port	9	Ports.
Slot	10	Slots.
Card	11	Cards.
LineNode	12	Line nodes.

4.119.3 See Also

NetTerrain.WebApi Namespace

4.120 NodeTypeProperty Class

This class contains parameters of a node type property.

4.120.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.NodeTypeProperty

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.120.2 Syntax

C#

```
public class NodeTypeProperty
```

The **NodeTypeProperty** type exposes the following members.

4.120.3 Constructors

	Name	Description
	NodeTypeProperty	

4.120.4 Fields

	Name	Description
	Angle	Angle for a property displayed text (0..360).
	Bold	If set to true makes the displayed property text bold.
	DefaultValue	Default value of the property.
	Displayed	If set to true sets the property as displayed on a diagram.
	FillColor	Text background hex color code for a displayed property (Example: '#000000').
	FontColor	Text font hex color code for a displayed property (Example: '#000000').
	FontFamily	Font family for a displayed property.

	Name	Description
🔷	FontSize	Text font size for a displayed property.
🔷	Id	Property id.
🔷	IsInProperties	If set to true the property is shown in the properties list when a node of this type is selected on a diagram.
🔷	IsSystem	If set to true indicates that the property belongs to system properties and cannot be deleted.
🔷	IsTypeField	If true indicates that this property is a reference to a host node type.
🔷	IsUniqueForAllTypes	If set to true makes a property unique for all node types.
🔷	IsUniqueForThisType	If set to true makes a property unique for this node type.
🔷	Italic	If set to true makes displayed property text italic.
🔷	Justification	Text justification mode for a displayed property.
🔷	LockList	If set to true locks the list of values if the property has it.
🔷	Mandatory	If set to true makes the property mandatory.
🔷	Name	Property name.
🔷	NodeTypeId	Node type id.
🔷	NotEditable	If set to true makes the property non-editable.
🔷	OffsetX	Text offset X for a displayed property.
🔷	OffsetY	Text offset Y for a displayed property.
🔷	Position	Property position number in a list of properties of this node type.
🔷	TextAlign	Text alignment for a displayed property.
🔷	Underline	If set to true makes displayed property text underlined.

4.120.5 See Also

NetTerrain.WebApi Namespace

4.121 OverrideRules Enumeration

This enumeration type sets override rules for node and link properties.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.121.1 Syntax

C#

```
public enum OverrideRules
```

4.121.2 Members

Member name	Value	Description
Equals	0	A property value is equal to an override value.
Contains	1	A property value contains an override value.
GreaterThan	2	A property value is greater than an override value.
LowerThan	3	A property value is lower than an override value.

4.121.3 See Also

NetTerrain.WebApi Namespace

4.122 Roles Enumeration

This enumeration type sets group roles.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.122.1 Syntax

C#

```
public enum Roles
```

4.122.2 Members

Member name	Value	Description
NoAccess	0	No access.
ReadOnly	1	Read Only.
DiagramReadOnly	2	Diagram Read Only.
Annotator	3	Annotator.
Updater	4	Updater.
Editor	5	Editor.
PowerUser	6	Power User.
Admin	7	Admin.

4.122.3 See Also

NetTerrain.WebApi Namespace

4.123 TextAligns Enumeration

This enumeration type sets horizontal alignment for multiline text within a field.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.123.1 Syntax

C#

```
public enum TextAligns
```

4.123.2 Members

Member name	Value	Description
Left	0	Left.
Center	1	Center.
Right	2	Right.

4.123.3 See Also

NetTerrain.WebApi Namespace

4.124 TextJustification Enumeration

This enumeration type sets text field justification point.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.124.1 Syntax

C#

```
public enum TextJustification
```

4.124.2 Members

Member name	Value	Description
TopLeft	0	Top Left corner of the text field.
Center	1	Center of the text field.

4.124.3 See Also

NetTerrain.WebApi Namespace

4.125 UpwardsPropagations Enumeration

This enumeration type sets upwards propagation of effects applied to an overriden instance.

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.125.1 Syntax

C#

```
public enum UpwardsPropagations
```

4.125.2 Members

Member name	Value	Description
NoPropagations	0	No Propagations.
ParentOnly	1	Parent Only.
ParentAndGrandparentOnly	2	Parent And Grandparent Only.
TopLevelOnly	3	Top Level Only.
AllLevels	4	All Levels.

4.125.3 See Also

NetTerrain.WebApi Namespace

4.126 User Class

This class contains parameters of a user.

4.126.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.User

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.126.2 Syntax

C#

```
public class User
```

The **User** type exposes the following members.

4.126.3 Constructors

	Name	Description
	User	

4.126.4 Fields

	Name	Description
	Comments	Additional comments.
	Description	User description.
	Email	User email address.
	GroupId	User group id. Ignored for Active Directory users when IsOverrideAdGroup parameter is set to false.
	Id	User id.
	IsAdAccount	If set to true indicates that the user belongs to an Active Directory domain.
	IsLocked	If set to true the user is locked.
	IsOverrideAdGroup	If set to true the Active Directory user uses an overridden group instead of the group determined by its domain controller. In this case the user's group is determined by the GroupId parameter as for native (non-AD) users.
	Name	User name.

4.126.5 See Also

NetTerrain.WebApi Namespace

4.127 Vendor Class

This class contains parameters of a vendor. Applicable for node types such as Racks, Devices or Cards.

4.127.1 Inheritance Hierarchy

System.Object   NetTerrain.WebApi.Vendor

Namespace: NetTerrain.WebApi **Assembly:** NetTerrain (in NetTerrain.dll) Version: 7.1.720

4.127.2 Syntax

C#

```
public class Vendor
```

The **Vendor** type exposes the following members.

4.127.3 Constructors

	Name	Description
◆	Vendor	

4.127.4 Fields

	Name	Description
◆	Id	Vendor id.
◆	Name	Vendor name.

4.127.5 See Also

NetTerrain.WebApi Namespace

This guide is part of the official documentation for netTerrain,
developed by Graphical Networks.

GRAPHICAL
NETWORKS

For more information, please visit
www.graphicalnetworks.com