Table of Contents

Overview

This reference specifies the REST service that Revit Server offers for server administration. It consists of two groups of APIs:

1. Information querying, which includes:
- Querying the properties of the server.
- Querying the contents of a specified folder.
- Querying the directory information of a specified folder.
- Querying the history of a specified model.
- Querying the file information of a specified model.
- Querying the project information of a specified model.
- Querying the thumbnail of a specified model.

2. Data managing, which includes:
- Locking and unlocking the entire server, a specified folder including all its descendents, or a specified model.
- Querying and unlocking a specified folder's descendents.
- Copying or moving a specified folder or a specified model.
- Creating a new folder.
- Renaming a specified folder or model.
- Deleting a specified folder or model.

Versions

The version of Revit Server specified in this reference is 2016.

Glossary

- Object: an object refers to the server, a folder, or a model.

Common Behaviors

This section specifies constraints that apply to all REST APIs.

Transport Protocol

All APIs are based on the Hypertext Transfer Protocol, version 1.1 (RFC 2616).

Media Type

Request bodies and response bodies are normally encoded in JSON, as specified in RFC4627.

Since JSON lacks time literal, so the date and time values are encoded to strings in format "\/Date(*ticks*)\/", where *ticks* represents milliseconds since epoch (UTC).

Base URI

> http://<*host*>/RevitServerAdminRESTService2016/AdminRESTService.svc

- <*host*>: the name or IP address of the server.

Object Path

In most APIs, object paths are included as segments or parameters in the URLs. All paths are relative to the server's root.

Since slashes "\" and "/" are URL's special characters, object paths are formatted as:
- server's root: *"|"*
- Folder Path: *"folderName1[|folderName2[…]]"*
- Model Path: *"[folderName1[|folderName2[…]|]]modelName.rvt"*

Query String

Some APIs require query strings. These strings are case-sensitive. Invalid parameter names will result in "405 Method Not Allowed" errors, and invalid parameter values will result in "400 Bad Request" errors.

Common Request Headers

Header Name	Description	Required
User-Name	Specifies the user name of the client. It is used to form an application-level session token. Supported value: *string*	Yes
User-Machine-Name	Specifies the machine name of the client. It is used to form an application-level session token. Supported value: *string*	Yes
Operation-GUID	Specifies a GUID for the request. It is used for server-side logging and diagnostics, so a unique GUID for every request is preferred. Supported value: *GUID in string format*	Yes

Common Response Headers

Header Name	Description	Required
Content-Length	Describes the length in bytes of the response body. Supported value: *integer in string format*	Yes

Content-Type	Describes the representation and syntax of the response body. Supported value: *application/json, image/png* Condition: Required by the APIs that return data.	Conditional
Location	Returns a new URI that can be used to request a representation of the newly created object. Supported value: *absolute URI*. Condition: Required by the APIs that create new objects.	Conditional

Common Status Codes

Status Code	Description
200 OK	The request has succeeded.
201 Created	The request has been fulfilled and resulted in a new object being created.
400 Bad Request	The request could not be understood by the server due to incorrect syntax or invalid parameter values.
404 Not Found	The object specified in the request's URL could not be found.
405 Method Not Allowed	The method (GET, PUT, DELETE, POST) specified in the request is not allowed for the request's URL.
414 Request-URI Too Long	Path of the object specified in the request's URL is longer than the server supports.
500 Internal Server Error	The server encountered an unexpected condition which prevented it from fulfilling the request.
501 Not Implemented	The server does not support the functionality required to fulfill the request.
503 Service Unavailable	The server is currently unable to handle the request due to a temporary overloading or maintenance of the server.

Information Querying APIs

This section specifies APIs that are used to query information about a server's properties and the data on the server.

GET /serverProperties

Queries the server's properties.

Request

URL

```
GET /serverProperties
```

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

```
{
    " MaximumFolderPathLength": int,
    " MaximumModelNameLength": int,
    "ServerRoles": [int],
    "Servers": [string]
}
```

- MaximumFolderPathLength: the maximum folder path length that the server supports.
- MaximumModelNameLength: the maximum model path length that the server supports.
- ServerRoles: the list of roles current server plays (0: host; 1: accelerator; 2: admin).
- Servers: the name list of servers (not including accelerators) in the Revit Server Network.

Response Status Codes

The implementation of this API uses only status codes common to all APIs. For more information, see Common Status Codes.

Example

The following example queries the properties of server Earth. The maximum lengths of folder path and model name it supports are 119 and 40 respectively. The server plays the roles of host and admin. The Revit Server Network comprises four servers: Mercury, Venus, Earth and Mars.

Request:

```
GET /serverProperties

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 120
Content-Type: application/json; charset=utf-8

{
    "MaximumFolderPathLength":119,
    "MaximumModelNameLength":40,
    "ServerRoles":[0,2],
    "Servers":[Mercury, Venus, Earth, Mars]
}
```

GET /{folderPath}/contents

Queries the contents of a folder.

Request

URL

GET /{*folderPath*}/contents

- *FolderPath*: the path of the specified folder.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

```
{
   "Path": string,
   "LockContext": string, "LockState": int,
   "ModelLocksInProgress":
   [
      {
         "Age": timespan, "ModelLockOptions": int,  "ModelLockType": int,
         "ModelPath": string, "TimeStamp":" datetime, "UserName": string
      }
   ]
   "Folders":
   [
      {
         "HasContents": boolean, "LockContext": string, "LockState": int,
         "ModelLocksInProgress": object array as above,
         "Name": string, "Size": int
      }
   ],
   "Models":
   [
      {
         "LockContext": string, "LockState": int, "ModelSize": int,
         "ModelLocksInProgress": object array as above,
         "Name": string, "ProductVersion": int, "SupportSize": int
      }
   ],
```

8

```
    "DriveFreeSpace": int, "DriveSpace": int
}
```

- Path: the folder path.
- LockContext: the context of the admin lock on the folder/model, describing the use of the admin lock such as copying or moving a folder from one server to another.
- LockState: the lock state of the folder/model (0 - unlocked; 1- locked; 2- has a locked ancestor; 3 – has one or more locked descendent; 4 – being unlocked; 5 – being locked).
- ModelLocksInProgress: the list of descendant models that are locked by the Revit clients.
- Age: the age of the model lock.
- ModelLockOptions: the combination of the model lock options (1-read; 2-write; 128-NonExclusiveReadOrWrite).
- ModelLockType: the type of the model lock (0-data; 1-permissions).
- ModelPath: the model path.
- TimeStamp: the time stamp of the model lock.
- UserName: the user who locks the model.
- Folders: the list of sub-folders.
- HasContents: whether the folder has any contents.
- Name: the name of folder/model.
- Size: the size in bytes of the folder.
- Models: the list of sub-models.
- ModelSize: the size in bytes of the model.
- ProductVersion: the version number of Revit that the model is lasted modified with.
- SupportSize: the size in bytes of the auxiliary data (such as user temporary data) for the model.
- DriveSpace: the total space in bytes of the drive where the folder exists.
- DriveFreeSpace: the free space in bytes of the drive where the folder exists.

Response Status Codes

The implementation of this API uses only status codes common to all APIs. For more information, see Common Status Codes.

Example

The following example queries the contents of 00Folder, which has a sub-folder 01Folder, and two models: 01CentralModel.rvt and 01LinkedModel.rvt.

Request:

```
GET /00Folder/contents

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 898
Content-Type: application/json; charset=utf-8

{
   "Path":"00Folder",
   "LockContext":null,"LockState":0,
   "ModelLocksInProgress":
   [
      {
         "Age":"PT3H54M3.5096034S","ModelLockOptions":131,"ModelLockType":0,
         "ModelPath":"00Folder\\01CentralModel.rvt",
         "TimeStamp":"\/Date(1323052799000)\/","UserName":"tim"
      }
   ],
   "Folders":
   [
      {
         "HasContents":true,"LockContext":null,"LockState":0,
         "ModelLocksInProgress":null,
         "Name":"01Folder","Size":5003839
      }
   ],
   "Models":
   [
      {
         "LockContext":null,"LockState":0, "ModelSize":4999816,
         "ModelLocksInProgress":
         [
            {
               "Age":"PT3H54M3.5096034S","ModelLockOptions":131,"ModelLockType":0,
               "ModelPath":"00Folder\\01CentralModel.rvt",
               "TimeStamp":"\/Date(1323052799000)\/","UserName":"tim"
            }
         ]
         "Name": "01CentralModel.rvt","ProductVersion":2016,"SupportSize":4095
      },
      {
         "LockContext":null,"LockState":0,"ModelSize":4999792,
         "ModelLocksInProgress":null,
         "Name":"01LinkedModel.rvt", "ProductVersion":2016,"SupportSize":4023
      }
   ],
   "DriveFreeSpace":72842928128,"DriveSpace":258461396992
}
```

GET /{folderPath}/DirectoryInfo

Queries the folder directory information.

Request

URL

> GET /{*folderPath*}/DirectoryInfo

- *FolderPath*: the path of the specified folder.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

```
{
    "Path": string,
    "DateCreated": datetime,
    "DateModified": datetime,
    "LastModifiedBy": string,
    "Size": int,
    "ModelSize": int,
    "FolderCount": int,
    "ModelCount": int,
    "IsFolder": Boolean,
    "Exists": Boolean
}
```

- Path: the folder path.
- DateCreated: the creation time.
- DateModified: the last modified time.
- LastModifiedBy: the user who did the last modification.
- Size: the size of the folder.
- ModelSize: the model size if the folder is a model's folder.
- FolderCount: the count of sub-folders under the folder.
- ModelCount: the count of models under the folder.
- IsFolder: whether the folder is a normal folder and not a model's folder.
- Exists: whether the folder exists.

11

Response Status Codes

The implementation of this API uses only status codes common to all APIs. For more information, see Common Status Codes.

Example

The following example queries the directory information of 00Folder, which has one sub-folder and two models.

Request:

```
GET /00Folder/DirectoryInfo

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 212
Content-Type: application/json; charset=utf-8

{
    "Path":"00Folder",
    "DateCreated":"\/Date(1294907700265)\/",
    "DateModified":"\/Date(1294971083276)\/",
    "Exists":true,
    "FolderCount":1,
    "IsFolder":true,
    "LastModifiedBy":null,
    "ModelCount":2,
    "ModelSize":0,
    "Size":8137964
}
```

GET /{modelPath}/history

Queries the submission history of a model.

Request

URL

```
GET /{modelPath}/history
```

- *ModelPath*: the path of the specified model.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

```
{
    "Path": string,
    "Items":
    [
        {
            "Comment": string,
            "Date": datetime,
            "User": string,
            "VersionNumber": int,
            "ModelSize": int,
            "SupportSize": int
        }
    ]
}
```

- Path: the model path.
- Items: the list of a model's submission history.
- Date: the date and time the submission was made to the model.
- User: the user who made the submission.
- VersionNumber: the version of the model created by the submission.
- ModelSize: the size of the model at the time the submission was made.
- SupportSize: The size of the auxiliary data (such as user temporary data) for the model at the time the submission was made.

Response Status Codes

The implementation of this API uses only status codes common to all APIs. For more information, see Common Status Codes.

Example

The following example queries the submission history of 00Folder\01CentraModel.rvt, which has two versions that were saved by Alice and Bob respectively.

Request:

```
GET /00Folder|01CentraModel.rvt/history

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 281
Content-Type: application/json; charset=utf-8

{
   "Path":"00Folder\01CentraModel.rvt",
   "Items":
   [
     {
        "Comment":"",
        "Date":"\/Date(1294974276000)\/",
        "ModelSize":4023703,
        "SupportSize":3917,
        "User":"Alice",
        "VersionNumber":1
     },
     {
        "Comment":"",
        "Date":"\/Date(1294974347000)\/",
        "ModelSize":4030877,
        "SupportSize":3909,
        "User":"Bob",
        "VersionNumber":2
     }
   ]
}
```

GET /{modelPath}/modelInfo

Queries the file information of a model.

Request

URL

```
GET /{modelPath}/modelInfo
```

- *ModelPath*: the path of the specified model.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

```
{
    "Path": string,
    "DateCreated": datetime,
    "DateModified": datetime,
    "LastModifiedBy": string,
    "ModelGUID": GUID in string,
    "ModelSize": int,
    "SupportSize": int
}
```

- Path: the model path.
- DateCreated: the creation time.
- DateModified: the last modification time.
- LastModifiedBy: the user who did the last modification.
- ModelGUID: the GUID of the model.
- ModelSize: the size of the model.
- SupportSize: The size of the auxiliary data (such as user temporary data) for the model.

Response Status Codes

The implementation of this API uses only status codes common to all APIs. For more information, see Common Status Codes.

Remarks

For pre-2013 models, empty ModelGUID will be returned.

Example

The following example queries the information of model 00CentralModel.rvt.

Request:

```
GET / 00CentralModel.rvt/modelInfo

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 281
Content-Type: application/json; charset=utf-8

{
   "Path":"00CentralModel.rvt",
   "DateCreated":"\/Date(1320977094080)\/",
   "DateModified":"\/Date(1321330137000)\/",
   "LastModifiedBy":"Alice",
   "ModelGUID":"ee4ac20b-22d3-4de8-9060-ed363b97ade0",
   "ModelSize":4134630,"
   SupportSize":1573147
}
```

GET /{modelPath}/projectInfo

Queries the project information of a model.

Request

URL

```
GET /{modelPath}/projectInfo
```

- *ModelPath*: the path of the specified model.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

The project information is organized as a JSON array, in which each item is a JSON object, corresponding to a group of parameters. In the object, there is one pair named "A:title" describing the group name, and one or more pairs describing the parameters in the group.

```
[
    {
        "A:title": string,
        "parameter_name":
        {
            "#text": string,
            "@displayName": string,
            "@type": string,
            "@typeOfParameter": string
        }
        …
    }
]
```

- A:title: the group name.
- Parameter_name: the parameter name.
- #text: the parameter value.
- @displayName: the parameter's display name.
- @type: the parameter's type (system, custom, shared).
- @typeOfParameter: the parameter's value type (length, number, material, text, yes/no).

17

Response Status Codes

The implementation of this API uses only status codes common to all APIs. For more information, see Common Status Codes.

Remarks

For pre-2013 models, this API will always return "405 Method Not Allowed".

Example

The following example queries the project information of 00CentralModel.rvt.

Request:

```
GET / 00CentralModel.rvt/projectInfo

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 281
Content-Type: application/json; charset=utf-8

[
  {
    "A:title":"Identity Data",
    "Author":
    {
      "#text":"Bob", "@type":"system", "@typeOfParameter":"Text"
    },
    "Building_Name":
    {
      "#text":"ABC Head Quarter", "@displayName":"Building Name",
      "@type":"system", "@typeOfParameter":"Text"
    },
    "Workset":
    {
      "#text":"Project Info", "@type":"system"
    }
  },
  {"A:title":"Energy Analysis"},
  {
    "A:title":"Other",
    "Client_Name":
    {
      "#text":"ABC Inc.", "@displayName":"Client Name",
```

```
      "@type":"system", "@typeOfParameter":"Text"
    },
    "Project_Address":
    {
      "#text":"1 Central Avenue, Beijing, China", "@displayName":"Project Address",
      "@type":"system", "@typeOfParameter":"Text"
    },
    "Project_Issue_Date":
    {
      "#text":"01 JAN 2011", "@displayName":"Project Issue Date",
      "@type":"system", "@typeOfParameter":"Text"
    },
    "Project_Name":
    {
      "#text":"ABC Head Quarter", "@displayName":"Project Name",
      "@type":"system", "@typeOfParameter":"Text"
    },
    "Project_Number":
    {
      "#text":"2011PJ0001", "@displayName":"Project Number",
      "@type":"system", "@typeOfParameter":"Text"
    },
    "Project_Status":
    {
      "#text":"In Progress", "@displayName":"Project Status",
      "@type":"system", "@typeOfParameter":"Text"
    }
  }
]
```

GET /{modelPath}/thumbnail?width={width}&height={height}

Gets the thumbnail of a model.

Request

URL

> GET /{*modelPath*}/thumbnail?width={*width*}&height={*height*}

- *ModelPath*: the path of the specified model.
- *Width*: width of expected thumbnail.
- *Height*: height of expected thumbnail.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

Response body contains a PNG image.

Response Status Codes

The implementation of this API uses only status codes common to all APIs. For more information, see Common Status Codes.

Example

The following example gets a 128*128 thumbnail of 00Folder\01CentralModel.rvt.

Request:

```
GET /00Folder|01CentralModel.rvt/thumbnail?width=128&height=128

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 1041
Content-Type: image/png

A PNG image in 128*128
```

Data Managing APIs

This section specifies APIs that are used to manage data on the server.

Specific Status Codes

Status Code	Description
403 Forbidden	There was a lock conflict that prevented the request from being processed. A lock conflict could occur in following cases: - A lock has already been applied on the object specified in the request's URL, or on one of its ancestors or one of its descendents. - A data-managing request is being processed on the object specified in the request's URL, or one of its ancestors, or one of its descendents. All data-managing APIs except for lock-related ones have implicit locking and unlocking operations. Some APIs may allow certain types of lock conflict. See the APIs' "Remarks" section for details.

409 Conflict	The request could not be completed due to a conflict with the current state of the object specified in the request's URL. The conflict could occur in the following cases:
	- There is a Revit client's user lock on the object.
	- In "DELETE /{objectPath}" API:
	    a. Destination already exists.
	    b. Destination and source are the same.
	- In "POST /{folderPath}/descendent" API:
	    a. Destination folder is under source.
	    b. When **CopyIncrement** is not specified, destination folder is the parent of source.
	    c. When **Merge** is specified, destination object already exists and it is a model.
	    d. When **Replace** is specified, destination object doesn't exist.

PUT /{objectPath}/lock

Lock the server, a folder or a model.

Request

URL

> PUT /{*objectPath*}/lock

- *ObjectPath*: the path of the server, the specified folder or the specified model.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Status Codes

The implementation of this API uses status codes common to all APIs, and codes specific to data managing APIs. For more information, see Common Status Codes and data managing APIs' Specific Status Codes.

Remarks

This API is idempotent, which means locking an already-locked object is allowed.

If a lock has been applied on an object, all data-managing requests except for lock-related requests on the object, its ancestors, and its descendents will be forbidden. Revit clients will also be forbidden to read and write any models related to the object in this case.

Example

The following example locks 00Folder\01CentralModel.rvt.

Request:

> PUT /00Folder|01CentralModel.rvt/lock
>
> User-Name: Tester
> User-Machine-Name: TestMachine
> Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93

Response:

> HTTP/1.1 200 OK
> Content-Length: 0

DELETE /{objectPath}/lock?objectMustExist={objectMustExist}

Unlocks the server, a folder, or a model.

Request

URL

> DELETE /{*objectPath*}/lock?objectMustExist={*objectMustExist*}

- *ObjectPath*: the path of the server, the specified folder, or the specified model.
- *ObjectMustExist*: whether the folder or model must exist.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Status Codes

The implementation of this API uses status codes common to all APIs, and codes specific to data managing APIs. For more information, see Common Status Codes and data managing APIs' Specific Status Codes.

Remarks

Parameter *ObjectMustExist* is used in the case of removing the lock record of a deleted folder or model.

This API is idempotent, which means unlocking an already-unlocked object is allowed.

Example

The following example unlocks 00Folder\01CentralModel.rvt.

Request:

> DELETE /00Folder|01CentralModel.rvt/lock?objectMustExist=true
>
> User-Name: Tester
> User-Machine-Name: TestMachine
> Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93

Response:

> HTTP/1.1 200 OK
> Content-Length: 0

DELETE /{objectPath}/inProgressLock

Cancel the in-progress locking operation on the server, a folder, or a model.

Request

URL

DELETE /{*objectPath*}/inProgressLock

- *ObjectPath*: the path of the server, the specified folder, or the specified model.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Status Codes

The implementation of this API uses status codes common to all APIs, and codes specific to data managing APIs. For more information, see Common Status Codes and data managing APIs' Specific Status Codes.

Remarks

This API is idempotent, which means no error will be reported if the object has been locked or has been unlocked.

Example

The following example cancels the in-progress locking operation on 00Folder.

Request:

DELETE /00Folder/inProgressLock

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93

Response:

HTTP/1.1 200 OK
Content-Length: 0

GET /{folderPath}/descendent/locks

Gets the lock information of the descendents of a folder.

Request

URL

```
GET /{folderPath}/descendent/locks
```

- *FolderPath*: the path of the specified folder.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

```
{
    "Path": string,
    "Items": [string]
    "DescendentHasLockContext": boolean
}
```

- *Path*: the folder path.
- *Items*: the list of paths of locked descendents.
- *DescendentHasLockContext:* whether there is any locked descendents that has lock context describing the use of the admin lock such as copying or moving a folder from one server to another.

Response Status Codes

The implementation of this API uses only status codes common to all APIs. For more information, see Common Status Codes.

Example

The following example gets the lock information of 00Folder's descendents, in which 01CentralModel.rvt and 01Folder have been locked.

Request:

```
GET /00Folder/descendent/locks

User-Name: Tester
User-Machine-Name: TestMachine
```

Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93

Response:

```
HTTP/1.1 200 OK
Content-Length: 76
Content-Type: application/json; charset=utf-8

{
  "Path":"00Folder",
  "Items":
  [
     "00Folder\\01CentralModel.rvt",
     "00Folder\\01Folder"
  ]
}
```

DELETE /{folderPath}/descendent/locks

Unlocks all locks of the descendents of a folder.

Request

URL

DELETE /{*folderPath*}/descendent/locks

- *FolderPath*: the path of the specified folder.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Body

```
{
    "Path": string,
    "FailedItems": [string]
}
```

- *Path*: the folder path.
- *FailedItems*: the list of paths of descendents that are not unlocked.

Response Status Codes

The implementation of this API uses status codes common to all APIs, and codes specific to data managing APIs. For more information, see Common Status Codes and data managing APIs' Specific Status Codes.

Example

The following example unlocks all locks of 00Folder's descendents.

Request:

```
DELETE /00Folder/descendent/locks

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 38
Content-Type: application/json; charset=utf-8

{
  "Path":"00Folder",
  "FailedItems":null
}
```

PUT /{folderPath}

Creates a new folder.

Request

URL

PUT /{*folderPath*}

- *FolderPath*: the path of the new folder to be created.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Status Codes

The implementation of this API uses status codes common to all APIs, and codes specific to data managing APIs. For more information, see Common Status Codes and data managing APIs' Specific Status Codes.

Remarks

As a prerequisite of the reques, the folder's parent folder must exist.

This API is idempotent, which means no error is reported if the folder already exists. "200 OK" instead of "201 Created" is returned in this case.

Example

The following example creates 01Folder under 00Folder.

Request:

PUT /00Folder|01Folder

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93

Response:

HTTP/1.1 201 Created
Content-Length: 0
Location: http://*host*/RevitServerAdminRESTService2016/00Folder|01Folder

DELETE /{objectPath}?newObjectName={newObjectName}

Rename or delete a folder or model.

Request

URL

```
DELETE /{objectPath}?newObjectName={newObjectName}
```

- *ObjectPath*: the path of the specified folder or model.
- *NewObjectName*: new name for the folder or model. Empty value means the object will be deleted.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Status Codes

The implementation of this API uses status codes common to all APIs, and codes specific to data managing APIs. For more information, see Common Status Codes and data managing APIs' Specific Status Codes.

Remarks

Deletion is recursive and undoable, so please be cautious in using this operation.

If a model or one of its ancestors is renamed, all the corresponding local models on Revit clients will become invalid, even if you then rename the central model with its original name.

Example

The following example renames 00Folder\01Folder to 00Folder\01Folder_renamed.

Request:

```
DELETE /00Folder|01Folder?newObjectName=01Folder_renamed

User-Name: Tester
User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93
```

Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Location: http://host/RevitServerAdminRESTService2016/ 00Folder|01Folder_renamed
```

POST /{sourceObjectPath}?destinationObjectPath={destinationObjectPath}&pasteAction={pasteAction}&replaceExisting={replaceExisting}

Copies or moves a folder or a model to another folder or model.

Request

URL

> POST
> /{sourceObjectPath}?destinationObjectPath={*destinationObjectPath*}&pasteAction={*pasteAction*}&replaceExisting={*replaceExisting*}

- *SourceObjectPath*: the path of the source object.
- *DestinationObjectPath*: the path of the destination object.
- *PasteAction*: the action type, which must be one of following values:

Value	Description
Copy	Copy the source object to the destination folder.
Move	Move the source object to the destination folder.

- *ReplaceExisting*: true if to replace destination object if it already exists, otherwise false.

Request Headers

The implementation of this API uses only request headers common to all APIs. For more information, see Common Request Headers.

Response

Response Headers

The implementation of this API uses only response headers common to all APIs. For more information, see Common Response Headers.

Response Status Codes

The implementation of this API uses status codes common to all APIs, and codes specific to data managing APIs. For more information, see Common Status Codes and data managing APIs' Specific Status Codes.

Example

The following example move 00Folder\01CentralModel.rvt to 10Folder.

Request:

> POST
> /10Folder|01CentralModel.rvt?destinationObjectPath=00Folder|01CentralModel.rvt&pasteAction=Move&replaceExisting=false
>
> User-Name: Tester

User-Machine-Name: TestMachine
Operation-GUID: 45FB8158-8BE3-43E5-9DFA-318BDBCB4C93

Response:

HTTP/1.1 201 Created
Content-Length: 0
Location: http://*host*/RevitServerAdminRESTService2016/10Folder|01CentralModel.rvt