Bravo Core Library

Bravo Consulting group | 11260 Roger Bacon Dr. Suite 503, Reston VA 20190

Developing in SharePoint 2013

Gunjan Datta

2015

# Overview

This document will give an overview of Bravo’s core library. This library is a tool developed in JavaScript which is utilized for developing applications for SharePoint 2013.

# Background

The core library utilizes the Representational State Transfer (REST) service that is comparable to SharePoint client object models. There are four ways to develop in SharePoint 2013:

* SSOM – Server Side Object Model
* CSOM – Client Side Object Model
* JSOM – JavaScript Object Model
* REST – Representational State Transfer

Both SSOM and CSOM are developed in C#, while JSOM is developed in JavaScript. The REST service allows developers to interact with SharePoint by web requests.

The core library was developed to serve as the REST service object model, which allows developers to develop against the service in JavaScript. Essentially, it takes the xml response of the REST service and transforms it into a JavaScript object.

# Why the Need?

Since Microsoft has already provided object models to develop in, the first question asked about this library is “Why do we need it?”. While JSOM is available, limitations can occur if the need to make synchronous calls to the server are needed in the code. For example, when developing JSLinks and applying customizations to list forms, JSOM cannot be used since its execution is asynchronous only. Another benefit of utilizing this library, is the ability develop code similar to the server side object model.

# How to Develop

The REST service has two main entry points, as shown below in Figure 1:

* [Site URL]/\_api/site
* [Site URL]/\_api/web

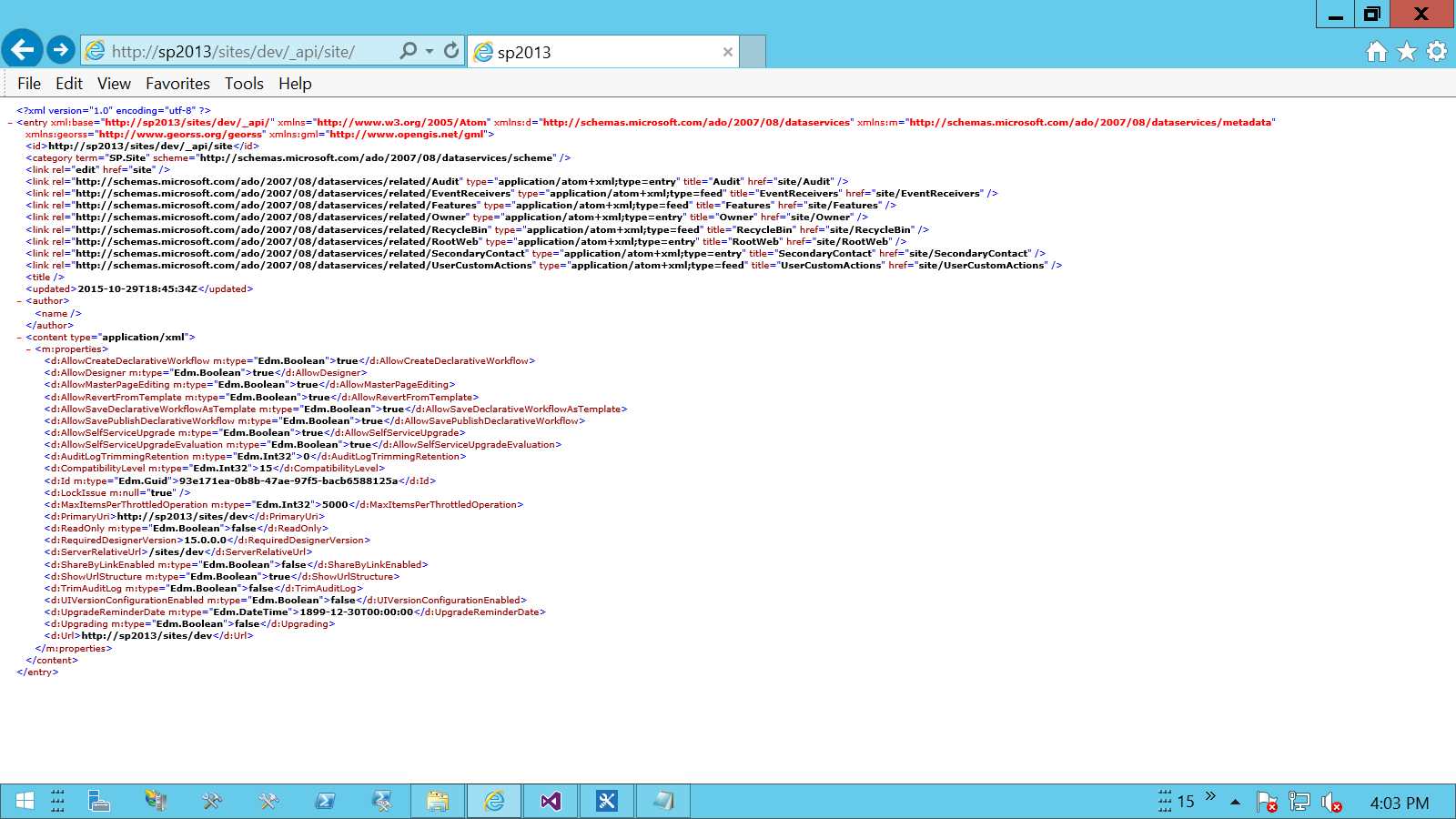


Figure - REST API Site XML Representation

The core library also has similar entry points, and allows the developer to create site and web objects that contain the methods and properties. An example of getting the current site is shown in Figure 2.

* var site = new BRAVO.Core.Site();
* var web = new BRAVO.Core.Web();

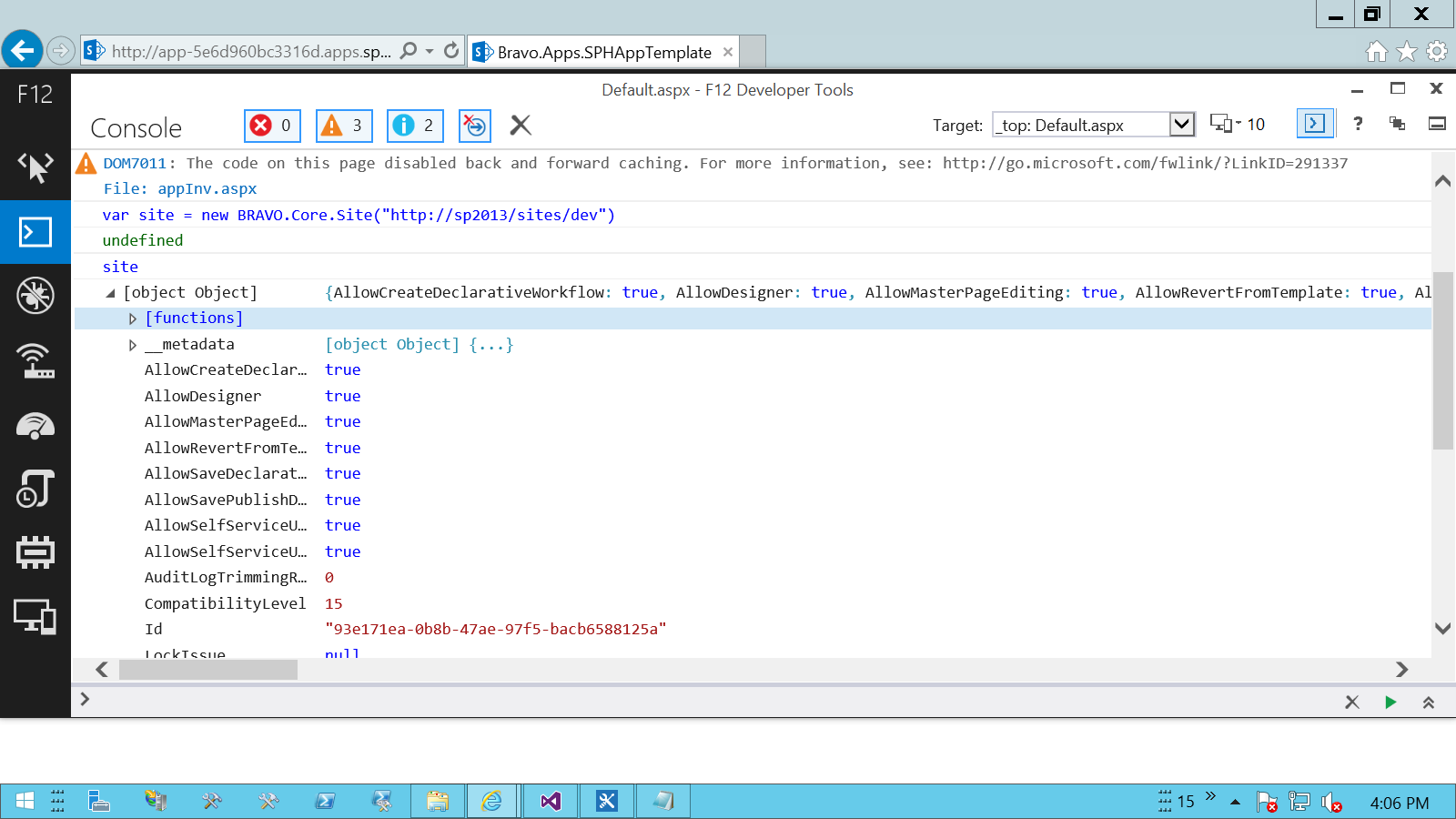


Figure - Bravo Core Library Site Object Representation

We’ve also inserted custom methods to help make life easier for developers, refer Methods/Properties Reference for additional details. We encourage new requests, bugs, issues, comments, etc on this library at our office community site.

## Asynchronous Requests

The core library also allows for asynchronous requests to be enabled. To create an asynchronous web/site object, the command is similar to the synchronous request:

* new BRAVO.Core.SiteAsync().done(function(site) { … });
* new BRAVO.Core.WebAsync().done(function(web) { … });

When the asynchronous flag is enabled, all requests will return a “Promise” object. The promise object allows the developer to pass a callback function to execute after the request is made.

## Switching Between Synchronous and Asynchronous Requests

The developer is able to switch between asynchronous and synchronous requests by setting the “asyncFl” property of the object to “true” or “false”.

# Other Uses

A major use of this library, is the ability to have powershell like access within the browser console window. As long as a reference to the library exists, the F-12 developer tools will allow the user to interact with the core library in a command-like interface.

# Blank ASPX Page Example

Below is sample code of a blank ASPX file. The code shown in **red**, is required for loading the library. The code shown in **green**, is an example of how to ensure the dependent SharePoint scripts are loaded.

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<%@ Page Language="C#" %>

<%@ Register tagprefix="SharePoint" namespace="Microsoft.SharePoint.WebControls" assembly="Microsoft.SharePoint, Version=15.0.0.0, Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>

<html dir="ltr" xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<meta name="WebPartPageExpansion" content="full" />

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<meta http-equiv="X-UA-Compatible" content="IE=10" />

<title>Dev Page</title>

<SharePoint:CssRegistration Name="default" runat="server"/>

**<SharePoint:ScriptLink runat="server" Localizable="False" Name="~sitecollection/style library/bravo/js/bravo.min.js"/>**

**<script type="text/javascript">**

**BRAVO.Core.loadDependencies(function() {**

**var web = new BRAVO.Core.Web();**

**if(web.exists) {**

**document.body.innerHTML += "<h3>Web Title: " + web.Title + "</h3>";**

**}**

**else { console.error("Unable to load core web"); }**

**});**

**</script>**

</head>

<body><form id="form1" runat="server">

**<SharePoint:FormDigest ID="FormDigest1" runat="server"></SharePoint:FormDigest>**

</form></body>

</html>

# Available Object Types

Currently, the core library allows you to interact with the following objects, through its api:

* List
* Search
* Site
* Social
  + Social Manager
* User Profile
  + People Manager
  + Profile Loader
* Web

# Default Methods

Refer to <https://msdn.microsoft.com/en-us/library/office/jj860569.aspx> for the REST service reference and examples. By default, all object types will have a common list of methods. Below is a list of these methods, and the methods available for the available object types.

## executeGet

The generic get method execution call.

### Input Parameters

* funcName – The function name.
* args – The function parameters passed in the function call.
* data – The function parameters passed as a data object.
* sendDataInBodyFl – Flag to send the data in the request body.
* bufferFl – Flag to indicate the response is an array buffer.
* metadataType – The metadata type.

## executePost

The generic post method execution call.

### Input Parameters

* funcName – The function name.
* args – The function parameters passed in the function call.
* data – The function parameters passed as a data object.
* sendDataInBodyFl – Flag to send the data in the request body.
* metadataType – The metadata type.
* method – The method type.

## getById

This method is only available to the collection object types. The generic method to search the results array and match the input parameter to the ‘Id’ property.

### Input Parameters

* id – The id to find.

## getByTitle

This method is only available to the collection object types. The generic method to search the results array and match the input parameter to the value of one of the following properties:

1. Title
2. Name
3. InternalName
4. StaticName

### Input Parameters

* title – The title to find.

## refresh

This method will execute the same request, and return it.

## setProperty

This method will update the property to the value passed in.

### Input Parameters

* name – The name of the property.
* value – The value to set the property to.

# Search Service Methods

## query

This method will execute a query against the search service.

### Input Parameters

* queryText – A string or object, representing the query parameters.
  + string – Executes a “GET” request, passing parameters with the url of the request.
    - “querytext=’Search query text goes here”
  + object – Executes a “POST” request, passing parameters with the body of the request.
    - { Querytext: “Search query text goes here” }

The search service allows for both GET and POST requests. An object should be used, when the query is complex or exceeds the limit of the URL length. Below are a list of query parameters:

* BypassResultTypes – A Boolean value, representing whether to perform result type processing for the query.
* ClientType – A string value, representing the client type that issued the query.
* CollapseSpecification – A string value, representing the managed properties that are used to determine how to collapse individual search results. Results are collapsed into one or a specified number of results if they match any of the individual collapse specifications. Within a single sollapse specification, results are collapsed if their properties match all individual properties in the collapse specification.
* Culture – An integer value, representing the locale id (LCID) for the query.
* DesiredSnippetLength – An integer value, representing the preferred number of characters to display in the hit-highlighted summary generated for a search result.
* EnableFql – A Boolean value, representing whether the query uses the FAST Query Language (FQL).
  + Default value is false
* EnableInterleaving – A Boolean value, representing whether the result tables that are returned for the result block are mixed with the result tables that are returned for the original query.
  + Default value is true
* EnableNicknames – A Boolean value, representing whether the exact terms in the search query are used to find matches, or if nicknames are used also.
  + Default value is false
* EnableOrderingHitHighlightedProperty – A Boolean value, representing whether the hit highlighted properties can be ordered.
* EnablePhonetic – A Boolean value, representing whether the phonetic forms of the query terms are used to find matches.
  + Default value is false
* EnableQueryRules – A Boolean value, representing whether to enable query rules for the query.
  + Default value is true
* EnableSorting – A Boolean value, representing whether to sort search results.
* EnableStemming – A Boolean value, representing whether stemming is enabled.
  + Default value is true
* GenerateBlockRankLog – A Boolean value, representing whether to return block rank log information in the BlockRankLog property of the interleaved result table. A block rank log contains the textual information on the block score and the documents that were de-duplicated.
* HiddenConstraints – A string value, representing additional query terms to append to the query.
* HitHighlightedMultivaluePropertyLimit – An integer value, representing the number of properties to show hit highlighting for in the search results.
* HitHighlightedProperties – A string or object, representing the properties to highlight in the search result summary when the property value matches the search terms entered by the user.
  + string – “hithighlightedproperties=’Title’”
  + object – { results: [ “Title” ] }
* MaxSnippetLength – An integer value, representing the number of characters to display in the hit-highlighted summary generated for a search result.
* PersonalizationData – A string value, representing the GUID for the user who submitted the search query.
* ProcessBestBets – A Boolean value, representing whether to return best bet results for the query.
  + *Note – This parameter is used only when EnableQueryRules is set to true, otherwise it will be ignored.*
* ProcessPersonalFavorites – A Boolean value, representing whether to return personal favorites with the search results.
* Properties – A string or object, representing the properties for the query.
  + string – “properties=’termid:guid’”
  + object – { results: [ { Name: “Boolean Property Name”, Value: { BoolVal: “True”, QueryPropertyValueTypeIndex: 3 } }, { Name: “Integer Property Name”, Value { IntVal: “1234”, QueryPropertyValueTypeIndex: 2 } } ] }
* Querytag – See “Querytext” property.
* QueryTemplatePropertiesUrl – A string value, representing the location of the queryparametertemplate.xml file. This file is used to enable anonymous users to make Search REST queries.
* Querytext – A string, representing the text for the search query.
* Querytemplate – A string value, representing the text that replaces the query text, as part of a query transform.
* RankingModelId – A string value, representing the id of the ranking model to use for the query.
* RefinementFilters – A string or object, representing the set of refinement filters used when issuing a refinement query.
  + string – “refinementfilters=fileExtension:equals(‘docx’), [*Filter 2*]”
  + object – { results: [ “fileExtension:equals(‘docx’)”, [*Filter 2*] ]
* Refiners – A string or object, representing the set of refiners to return in a search result.
  + string – “refiners=’*Refiner 1, Refiner 2*’”
  + object – { results: [ ‘*Refiner 1*’, ‘*Refiner 2*’ ]
* ReorderingRules – An object, representing special rules for reordering search results. These rules can specify that documents matching certain conditions are ranked higher or lower in the results. This property applies only when search results are sorted based on rank.
  + { results: [ { MatchValue: “*Some Value*”, Boost: “10”, MatchType: “0” } ] }
* ResultsURL – A string value, representing the url for the search results page.
* RowLimit – An integer value, representing the maximum number of rows overall that are returned in the search results. Compared to *RowsPerPage, RowLimit* is the maximum number of rows returned overall.
* RowsPerPage – An integer value, representing the maximum number of rows to return per page. Compared to *RowLimit, RowsPerPage* refers to the maximum number of rows to return per page, and is used primarily when you want to implement paging for search results.
* SelectProperties – A string or object, representing an array of managed properties to return in the search results.
  + string – “selectproperties=’*Managed Property 1, Managed Property 2*’”
  + object – { results: [ *Managed Property 1, Managed Property 2* ] }
* SortList – A string or object, representing the list of properties by which the search results are ordered.
  + string – “sortlist=’rank:descending,modifiedby:ascending’”
  + object – { results: [ { Property: “Created”, Direction: 0 }, { Property: “FileExtension”, Direction: 1 } ] }
* SourceId – A string value, representing the source id to use for executing the search query.
* StartRow – An integer value, representing the first row that is included in the search results that are returned. You use this parameter when you want to implement paging for search results.
* SummaryLength – An integer value, representing the number of characters to display in the result summary for a search result.
* Timeout – An integer value, representing the time in milliseconds before the query request times out.
  + Default value is 30,000 ms (30 sec)
* TrimDuplicates – A Boolean value, representing whether duplicate items are removed from the search.
  + Default value is true
* UILanguage – An integer value, representing the locale identifier (LCID) of the user interface.

## suggest

This method will executes a query suggestion against the search service.

### Input Parameters

* query – A string value, representing the query parameters.

Below are a list of query parameters:

* Culture – An integer value, representing the locale id (LCID) for the query.
* EnableQueryRules – A Boolean value, representing whether to turn on query rules for this query.
  + Default value is true
* EnableStemming – A Boolean value, representing whether stemming is enabled.
  + Default value is true
* fCapitalizeFirstLetters – A Boolean value, representing whether to capitalize the first letter in each term in the returned query suggestions. True to capitalize the first letter in each term.
  + Default value is false
* fHitHighlighting – A Boolean value, representing whether to hit-highlight or format in bold the query suggestions. True will format in bold.
  + Default value is true
* fPrefixMatchAllTerms – A Boolean value, representing whether to return query suggestions for prefix matches.
* fPreQuerySuggestions – A Boolean value, representing whether to retrieve pre-query or post-query suggestions.
* iNumberOfQuerySuggestions – An integer value, representing the number of query suggestions to retrieve.
  + Default value is 5
* iNumberOfResultSuggestions – An integer value, representing the number of personal results to retrieve.
  + Default value is 5
* Querytext – A string, representing the text for the search query.
* ShowPeopleNameSuggestions – A Boolean value, representing whether to include people names in the returned query suggestions.
  + Default value is true

# SharePoint Object Methods

## SP.ContentType

### addFieldLink

Refer to field link collection add method.

### deleteObject

This method will delete the content type.

### getFieldByInternalName

This method will return a SP.Field object, by the internal name property.

#### Input Parameters

* name – A string value, representing the internal name of the field.

### getFieldByStaticName

This method will return a SP.Field object, by static name property.

#### Input Parameters

* name – A string value, representing the static name of the field.

### getFieldByTitle

This method will return a SP.Field object, by title property.

#### Input Parameters

* title – A string value, representing the display name of the field.

### getFieldLinkByName

This method will return a SP.FieldLink link object, by the name property.

#### Input Parameters

* name – A string value, representing the internal name of the field.

### update

This method will update the content type properties.

#### Input Parameters

* data – A data object, representing the SP.ContentType properties.

## SP.ContentTypeCollection

### add

This method will add a new content type to the collection.

#### Input Parameters

* data – A data object, representing the SP.ContentTypeCreationInformation properties.

#### Data Properties

* Description – A string value, representing the description of the content type.
* Group – A string value, representing the group to associate the content type to.
* Name – A string value, representing the name of the content type.
* ParentContentType – A string value, representing the parent of the content type.

### addAvailableContentType

This method will add an existing content type to the collection.

#### Input Parameters

* contentTypeId – A string value, representing the content type id.

### getById

This method will return a SP.ContentType object, by the content type id.

#### Input Parameters

* contentTypeId – A string value, representing the content type id.

## SP.Field

### deleteObject

This method will delete the field.

### setShowInDisplayForm

This method will update the ShowInDisplayForm property of the field.

#### Input Parameters

* A boolean value, to set the property to.

### setShowInEditForm

This method will update the ShowInEditForm property of the field.

#### Input Parameters

* A boolean value to, set the property to.

### setShowInNewForm

This method will update the ShowInNewForm property of the field.

#### Input Parameters

* A boolean value to, set the property to.

### update

This method will update the field properties.

#### Input Parameters

* data – A data object, representing the SP.Field properties.

## SP.FieldCollection

### addDependentLookup

This method will add a lookup field to the collection.

#### Input Parameters

* data - A data object, representing the function parameters.

#### Data Properties

* displayname – A string value, representing the title of the new field.
* primarylookupfieldid – A string value, representing the ID of the lookup field.
* showfield – A string value, representing the internal name of the field from the target list to display as the new field’s value.

### addField

This method will add a field to the collection.

#### Input Parameters

* data – A data object, representing the SP.FieldCreationInformation properties.

#### Data Properties

* Choices – A string value, representing the available choices for the SP.FieldChoice field type.
* FieldTypeKind – A string value, representing the type of field.
* IsCompactName – Indicates whether only the first eight characters are used for the field name.
* LookupFieldName – The name of the source lookup field.
* LookupListId – The ID of the target list for the source lookup field.
* LookupWebId – The ID of the web containing the target list.
* Required – Specifies whether the field is required.
* Title – Specifies the display name of the field.

### createFieldAsXml

This method will add a field to the collection, that is defined by its Schema XML.

#### Input Parameters

* data – A data object, representing the SP.XmlSchemaFieldCreationInformation properties.

#### Data Properties

* Options – The options to use to add the field.
* SchemaXml – The CAML string contains the field schema.

### getById

This method will return a SP.Field object, by Id.

#### Input Parameters

* A string value representing the field’s id.

### getByInternalNameOrTitle

This method will return a SP.Field object, by the DisplayName or InternalName property.

#### Input Parameters

* A string value representing the field’s display or internal name property.

### getByTitle

This method will return a SP.Field object, by the DisplayName property.

#### Input Parameters

* A string value representing the field’s display name property.

## SP.FieldLink

## SP.FieldLinkCollection

### add

This method will add a field link to the collection.

#### Input Parameters

* data – A data object, representing the SP.FieldLink properties.

#### Data Properties

* FieldInternalName – The internal name of the field.
* Hidden – Specifies whether the field is displayed in forms.
* Required – Specifies whether the field requires a value.

### getById

This method will return a SP.FieldLink object, by the Id property.

#### Input Parameters

* A string value representing the field link’s id property.

## SP.File

### approve

This method will approve the file submitted for content approval with the specified comment.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* comment – The comment for the approval.

### cancelUpload

This method will stop the chunk upload session without saving the uploaded data. If the file doesn’t already exist in the library, the partially uploaded file will be deleted.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* uploadId – The GUID of the upload session.

### checkIn

This method checks the file in to the document library, based on the check-in type.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* comment – A string value, representing the comment for the check-in.
* checkInType – An int32 value, representing the SP.CheckinType enumeration.
  + MinorCheckIn – 0
  + MajorCheckIn – 1
  + OverwriteCheckIn - 2

### checkOut

This method checks the file out of the document library.

### continueUpload

This method continues the chunk upload session with an additional fragment. The current file content is not changed.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* uploadId – The GUID of the upload session.
* fileOffset – The int64 size of the offset into the file where the fragment starts.

### copyTo

This method copies the file to the destination url.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* strNewUrl – The absolute or server relative url of the destination file path to copy to.
* bOverWrite – A Boolean value representing whether to overwrite a file with the same name in the same location.

### deleteObject

This method will delete the file.

### deny

This method denies approval for a file that was submitted for content approval.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* comment – The comment for the denial.

### finishUpload

This method uploads the last file fragment and commits the file. The current file content is changed when this method completes.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* uploadId – The GUID of the upload session.
* fileOffset – The size of the offset into the file where the fragment starts.

### getLimitedWebPartManager

This method returns a SP.WebParts.LimitedWebPartManager for the file.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* scope – The SP.WebParts.PersonalizationScope to get.
  + User – 0
  + Shared – 1

### moveTo

This method moves the file to the specified destination url.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* newUrl – The absolute or server relative url of the destination file path to move to.
* flags – The bitwise SP.MoveOperations value for how to move the file.
  + Overwrite – 1
  + AllowBrokenThickets – 8
    - Move even if supporting files are separated from the file.

### openBinaryStream

This method returns the file as a stream.

### publish

This method submits the file for content approval with the specified comment.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* comment – The comment for the published file.
  + Length must be <= 1023

### recycle

This method moves the file to the recycle bin, and returns the identifier of the new recycle bin item.

### saveBinaryStream

This method will send the file contents in the post body.

#### Input Parameters

* stream – The file stream in the post body.

### startUpload

This method starts a new chunk upload session and uploads the first fragment. The current file content is not changed when this method completes.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* uploadId – The GUID of the upload session

### undoCheckOut

This method reverts an existing checkout for the file.

### unpublish

This method removes the file from content approval or unpublishes a major version.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* comment – The comment for the unpublish operation.
  + Length must be <= 1023

### update

This method updates the file properties.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

Refer to the SP.File REST documentation.

## SP.FileCollection

### add

This method will add a file to the collection.

#### Input Parameters

* data – A data object, representing the function parameters.
* stream – The file stream in the post body.

#### Data Properties

* overwrite – A boolean value representing whether to overwrite a file with the same name in the same location.
* url – The folder relative url of the file.

### addTemplateFile

This method adds a ghosted file to an existing list or document library.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* urlOfFile – The server-relative url where you want to save the file.
* templateFileType – The SP.TemplateFileType to use to create the file.
  + StandardPage – 0
  + WikiPage – 1
  + FormPage – 2

### getByUrl

This method will get the file at the specified url.

#### Input Parameters

* A string representing the server-relative url of the file.

## SP.FileVersion

### deleteObject

This method will delete the file version.

## SP.FileVersionCollection

### deleteAll

This method will delete all the file version objects in the collection.

### deleteById

This method deletes the specified version of the file, by the specified id.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* vid – The ID of the file version to delete.

### deleteByLabel

This method deletes the specified version of the file, by the specified label.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* versionlabel – The label of the file version to delete.

### getById

This method returns a SP.FileVersion object, by the specified id.

#### Input Parameters

* An int32 value representing the id of the file version.

### restoreByLabel

This method creates a new file version from the file specified by the version label.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* versionlabel – The label of the file version.

## SP.Folder

### addFile

Refer to the SP.FileCollection addFile method.

### addSubFolder

Refer to the SP.FolderCollection add method.

### deleteObject

This method will delete the folder.

### getFile

This method will return a SP.File object, by the specified file name.

#### Input Parameters

* name – A string value representing the file name.

### getSubFolder

This method will return a SP.Folder object, by the specified folder name.

#### Input Parameters

* name – A string value representing the folder name.

### update

This method updates the folder properties.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

Refer to the SP.Folder REST documentation.

## SP.FolderCollection

### add

This method adds a folder to the collection.

#### Input Parameters

* url – A string value representing the path where you want to add the folder. This value can be represented as one of the following:
  + fully-qualified url
  + server-relative url
  + site-relative url

### getByUrl

This method returns a SP.Folder object, by the specified url.

#### Input Parameters

* A string value representing the server-relative url of the folder.

## SP.Group

### containsUser

This method returns a boolean value, representing whether the specified user belongs to this group.

#### Input Parameters

* A SP.User object representing the user.

## SP.GroupCollection

### getById

This method returns a SP.Group object, by the specified group id.

#### Input Parameters

* An int32 value, representing the group id.

### getByName

This method returns a SP.Group object, by the specified group name.

#### Input Parameters

* A string value, representing the group name.

### removeById

This method removes a group from the collection, by the specified group id.

#### Input Parameters

* An int32 value, representing the group id.

### removeByLoginName

This method removes a group from the collection, by the specified group’s login name property.

#### Input Parameters

* A string value, representing the group’s login name property.

## SP.LimitedWebPartManager

### Get\_WebParts

This method returns a SP.WebParts.WebPartDefinitionCollection object, representing the webparts contained in the page.

## SP.List

### addContentType

Refer to the SP.ContentTypeCollection add method.

### addExistingContentType

Refer to the SP.ContentTypeCollection addAvailableContentType method.

### addField

Refer to the SP.FieldCollection addField method.

### addFieldAsXml

Refer to the SP.FieldCollection createFieldAsXml method.

### addItem

This method adds a new item to the list item collection.

#### Input Parameters

* data – The data object containing key value pairs, representing the internal field name as the key.

### addSiteGroup

Refer to the SP.RoleAssignmentCollection addRoleAssignment method.

### addSubFolder

This method adds a sub-folder, by the specified sub-folder name.

#### Input Parameters

* name – A string value, representing the name of the sub-folder.

### addView

This method adds a view to the list view collection.

#### Input Parameters

* data – A data object representing the SP.View object properties.

#### Data Properties

Refer to the SP.View REST documentation

### breakRoleInheritance

This method creates unique role assignments for the list.

#### Input Parameters

* data – A data object containing the function parameters.

#### Data Properties

* copyroleassignments – A boolean value, representing whether to copy the role assignments from the web.
* clearsubscopes – A boolean value, representing whether to clear the child role assignments.

### deleteObject

This method will delete the list.

### getChanges

This method returns a SP.ChangeCollection object, representing the change log that have occurred within the list, by the specified query.

#### Input Parameters

* data – A data object representing a SP.ChangeQuery object.

#### Data Properties

Refer to the SP.ChangeQuery REST documentation.

### getContentType

This method returns a SP.ContentType object, based on the specified content type name.

#### Input Parameters

* name – A string value, representing the content type’s name property.

### getContentTypeById

This method returns a SP.ContentType object, based on the specified content type id.

#### Input Parameters

* id – A string value, representing the content type’s id property.

### getDefaultDisplayFormUrl

This method returns a string value, representing the url to the list’s display form.

### getDefaultEditFormUrl

This method returns a string value, representing the url to the list’s edit form.

### getDefaultNewFormUrl

This method returns a string value, representing the url to the list’s new form.

### getDefaultViewUrl

This method returns a string value, representing the url to the list’s default view.

### getField

This method returns a SP.Field object, based on specified title.

#### Input Parameters

* title – A string value, representing one of the field’s following properties:
  + Title
  + InternalName
  + StaticName

### getFieldById

This method returns a SP.Field object, based on the specified field id property.

#### Input Parameters

* id – A string value, representing the field’s id property.

### getFieldByInternalName

This method returns a SP.Field object, based on the specified field internal name property.

#### Input Parameters

* name – A string value, representing the field’s internal name property.

### getFieldByStaticName

This method returns a SP.Field object, based on the specified field static name property.

#### Input Parameters

* name – A string value, representing the field’s static name property.

### getFieldByTitle

This method returns a SP.Field object, based on the specified field title property.

#### Input Parameters

* title – A string value, representing the field’s title property.

### getItemById

This method returns a SP.ListItem object, based on the specified list item’s id property.

#### Input Parameters

* id – An int32 value, representing the list item’s id property.

### getItemByTitle

This method returns a SP.ListItem object, based on the specified field internal name property.

#### Input Parameters

* title – A string value, representing the list item’s title property.

### getItems

This method returns a SP.ListItemCollection object, based on the specified query.

#### Input Parameters

* data – A data object, representing the SP.CamlQuery.

#### Example

{ query: { \_\_metadata: { type: “SP.CamlQuery” }, ViewXml: “<View><Query> … </Query></View>” } }

#### Data Properties

Refer to the SP.CamlQuery REST documentation.

### getItemsByFilter

This method returns a SP.ListItemCollection object, based on the specified filter.

#### Input Parameters

* filter – A string value, representing the OData query operations.
  + *Ntote - This filter prepends “$filter=” to the input parameter*

### getListItemChangesSinceToken

This method returns a stream containing XML, representing the changes to the list items, based on the specified query.

#### Input Parameters

* data – A data object, representing the SP.ChangeLogItemQuery query.

#### Data Properties

Refer to the SP.ChangeLogItemQuery REST documentation.

### getRelatedFields

This method returns a SP.RelatedFieldCollection object, representing a collection of lookup fields.

### getSchemaXml

This method returns a string value, representing the list’s schema XML property.

### getSubFolder

This method returns a SP.Folder object, specified by the sub-folder name.

#### Input Parameters

* name – A string value, representing the sub-folder name.

### getUserEffectivePermissions

This method returns a SP.BasePermissions object, representing the set of permissions for the specified user.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* user – A string value, representing the login name of the user.

### getView

This method returns a SP.View object, by the specified view’s id property.

#### Input Parameters

* A string value, representing the view’s id property.

### getViewById

This method returns a SP.View object, by the specified view’s id property.

#### Input Parameters

* A string value, representing the view’s id property.

### getViewByTitle

This method returns a SP.View object, by the specified view’s name property.

#### Input Parameters

* A string value, representing the view’s name property.

### hasAccess

This method returns a boolean value, representing whether the specified user has the specified permissions.

#### Input Parameters

* userName – A string value, representing the user’s login name.
* permissions – An array, representing the SP.PermissionKind enumeration types.

### recycle

This method moves the list to the recycle bin, and returns the identifier of the new recycle bin item.

### renderListData

This method returns a string, representing the list data specified by the query.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* viewXml – A string value, representing the SP.View CAML query.

### renderListFormData

This method returns a string value, representing the field values and field schema attributes for the list item.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* itemid – An int32 value, representing the list item id.
* formid – A string value, representing form GUID.
* mode – An int32 value, representing the SP.ControlMode.
  + Display – 1
  + Edit – 2
  + New – 3

### reserveListItemId

This method returns an int32 value, representing the list item id to be reserved.

### resetRoleInheritance

This method resets the role inheritance, and inherits the role assignments for the list.

### update

This method will update the list properties.

#### Input Parameters

* data – A data object, representing the SP.List properties.

## SP.ListCollection

### ensureSiteAssetsLibrary

This method returns a SP.List object, representing the default asset location for images or other files, which the users upload to their wiki pages.

### ensureSitePagesLibrary

This method returns a SP.List object, representing the default location for wiki pages.

### getById

This method returns a SP.List object, by the specified list id.

#### Input Parameters

* id – A string value, representing the GUID of the list id.

### getByTitle

This method returns a SP.List object, by the specified list title.

#### Input Parameters

* title – A string value, representing the list’s title property.

## SP.ListItem

### breakRoleInheritance

This method creates unique role assignments for the list item.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* copyroleassignments – A boolean value, representing whether to copy the role assignments from the web.
* clearsubscopes – A boolean value, representing whether to clear the child role assignments.

### deleteObject

This method will delete the list item.

### getUserEffectivePermissions

This method returns a SP.BasePermissions object, representing the set of permissions for the specified user.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* user – A string value, representing the login name of the user.

### recycle

This method moves the list item to the recycle bin, and returns the identifier of the new recycle bin item.

### resetRoleInheritance

This method resets the role inheritance, and inherits the role assignments for the list item.

### update

This method will update the list item properties.

#### Input Parameters

* data – A data object containing the SP.ListItem properties.

### validateUpdateListItem

This method validates and sets the values of the specified collection of fields for the list item.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* formValues – An array, representing the fields to change and their new values.
* bNewDocumentUpdate – A boolean value, representing whether the list item is a document being updated after upload.

## SP.ListItemCollection

### getById

This method returns a SP.ListItem object, by the specified list item id.

#### Input Parameters

* id – An int32 value, representing the list item id.

## SP.RoleAssignment

### deleteObject

This method will delete the role assignment.

## SP.RoleAssignmentCollection

### addRoleAssignment

This method adds a new role assignment to the collection, by the specified principal id and role definition.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* principalid – An int32 value, representing the principal id.
* roledefid – An int32 value, representing the role definition id.

### getByPrincipalId

This method returns a SP.RoleAssignment object, by the specified principal id.

#### Input Parameters

* An int32 value, representing the role assignment’s principal id.

### removeRoleAssignment

This method removes a role assignment from the collection, by the specified principal id and role definition.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* principalid – An int32 value, representing the principal id.
* roledefid – An int32 value, representing the role definition id.

## SP.RoleDefinition

### deleteObject

This method will delete the role definition.

## SP.RoleDefinitionCollection

### getById

This method returns a SP.RoleDefinition object, by the specified id.

#### Input Parameters

* An int32 value, representing the role definition’s id.

### getByName

This method returns a SP.RoleDefinition object, by the specified name.

#### Input Parameters

* A string value, representing the role definition’s name.

### getByType

This method returns a SP.RoleDefinition object, by the specified id.

#### Input Parameters

* An int32 value, representing the Microsoft.SharePoint.Client.RoleType enumeration value.
  + None – 0
  + Guest – 1
  + Reader – 2
  + Contributor – 3
  + WebDesigner – 4
  + Administrator – 5
  + Editor – [MSDN documentation doesn’t specify a value for this]

## SP.Site

### addCustomAction

This method will add a user custom action to the site’s user custom action collection.

#### Input Parameters

* data – A data object representing the SP.UserCustomAction object properties.

#### Data Properties

Refer to the SP.UserCustomAction REST documentation.

### createPreviewSPSite

This method creates a temporary evaluation of the site, for the purposes of determining whether an upgrade is likely to be successful.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* upgrade – A boolean value, representing whether the evaluation site must be upgraded when it’s created.
* sendemail – A boolean value, representing whether to send a notification email to the requestor and site collection administrators, at the completion of the creation of the preview site.

### extendUpgradeReminderDate

This method extends the upgrade reminder date, by the web application’s upgrade reminder delay property.

### getCatalog

This method returns a SP.List object, by the specified catalog type.

#### Input Parameters

* An int32 value, representing the Microsoft.SharePoint.SPListTemplateType enumeration value.

### getChanges

This method returns a SP.ChangeCollection object, representing the change log that have occurred within the site, by the specified query.

#### Input Parameters

* data – A data object representing a SP.ChangeQuery object.

#### Data Properties

Refer to the SP.ChangeQuery REST documentation.

### getCustomAction

This method returns a SP.UserCustomAction object, by the specified title.

#### Input Parameters

* title – A string value, representing the user custom action’s title or name property.

### getCustomListTemplates

This method returns a SP.ListTemplateCollection object, by the specified web.

#### Input Parameters

* TBD – Need to figure out how the web object is passed to the REST call.

### getRootWeb

This method returns a SP.Web object, representing the root web of the site.

### getWebTemplates

This method returns a SP.WebTemplateCollection object, representing the available site definitions.

#### Input Parameters

* LCID – An uint32 object, representing the language of the site definition.
* overrideCompatLevel – An int32 object, representing the compatibility level of the site definition.

### hasAccess

This method returns a boolean value, representing whether the specified user has the specified permissions.

#### Input Parameters

* userName – A string value, representing the user’s login name.
* permissions – An array, representing the SP.PermissionKind enumeration types.

### invalidate

This method invalidates cached upgrade information about the site, so that this information will be recomputed the next time it is needed.

### needsUpgradeByType

This method returns a boolean value, representing if an upgrade is needed.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* versionUpgrade – A boolean value, representing whether a version-to-version site collection upgrade is requested.
* recursive – A boolean value, representing whether child upgradable objects will be inspected.

### openWeb

This method returns a SP.Web object, by the specified url of the web.

#### Input Parameters

* strUrl – A string value, representing the server or site relative url of the web.

### openWebById

This method returns a SP.Web object, by the specified id of the web.

#### Input Parameters

* gWebId – A string value, representing the web id property.

### runHealthCheck

This method runs a heath check, based on the specified parameters.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* ruleId – A string value, representing the GUID of the rule id.
* bRepair – A boolean value, representing whether the repairable rules are to be run in repair mode.
* bRunAlways – A boolean value, representing whether the rules will be run as a result of this call or cached results from a previous run.

### runUpgradeSiteSession

This method either runs or schedules a site collection upgrade, based on the specified parameters.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* versionUpgrade – A boolean value, representing whether a version-to-version site collection upgrade is requested.
* queueOnly – A boolean value, representing whether the upgrade is run immediately or queued for a later run.
* sendEmail – A boolean value, representing whether to send a notification email to the requestor and site collection administrators, at the completion of the upgrade.

### sendEmail

This method sends an email, by the specified parameters.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* body – A string value, representing the body of the email.
* from – A string value, representing who the email is sent from.
* subject – A string value, representing the subject of the email.
* to – An array object, representing the email addresses to send the email to.

### update

This method will update the site properties.

#### Input Parameters

* data – A data object containing the SP.Site properties.

### updateClientObjectModeUseRemoteAPIsPermissionSetting

This method sets whether the client-side object model (CSOM) requests that are made in the context of any site inside the site collection require UseRemoteAPIs permission.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* requireUseRemoteAPIs – A boolean value, representing whether the client-side object model (CSOM) requests that are made in the context of any site inside the site collection require UseRemoteAPIs permission.

## SP.UserCollection

### getByEmail

This method returns a SP.User object, by the specified email.

#### Input Parameters

* A string value, representing the user’s email address property.

### getById

This method returns a SP.User object, by the specified id.

#### Input Parameters

* An int32 value, representing the user’s id property.

### getByLoginName

This method returns a SP.User object, by the specified login name.

#### Input Parameters

* A string value, representing the user’s login name property.

### removeById

This method removes the user from the collection, by the specified id.

#### Input Parameters

* An int32 value, representing the user’s id property.

### removeByLoginName

This method removes the user from the collection, by the specified login name.

#### Input Parameters

* A string value, representing the user’s login name.

## SP.UserCustomAction

### deleteObject

This method will delete the user custom action.

## SP.UserCustomActionCollection

### clear

This method removes all user custom actions from the collection.

### getById

This method returns a SP.UserCustomAction object, specified by the id.

#### Input Parameters

* A string value, representing the user custom action’s id.

## SP.View

### deleteObject

This method will delete the list view.

### renderAsHtml

This method returns a string, representing the list view as HTML.

### update

This method will update the view properties.

#### Input Parameters

* data – A data object containing the SP.View properties.

## SP.ViewCollection

### getById

This method returns a SP.View object, by the specified id.

#### Input Parameters

* A string value, representing the view’s id property.

### getByTitle

This method returns a SP.View object, by the specified title.

#### Input Parameters

* A string value, representing the view’s name property.

## SP.ViewFieldCollection

### addViewField

This method adds a field to the collection, specified by the field’s internal name.

#### Input Parameters

* A string value, representing the field’s internal name property.

### moveViewFieldTo

This method moves a view field to a specified position in the collection.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* field – A string value, representing the field’s internal name property.
* index – An int32 value, representing the field’s position in the collection.

### removeAllViewFields

This method removes all view fields in the collection.

### removeViewField

This method removes a field from the collection, specified by the field’s internal name.

#### Input Parameters

* A string value, representing the field’s internal name property.

## SP.Web

### addContentType

Refer to the SP.ContentTypeCollection add method.

### addCustomAction

This method will add a user custom action to the site’s user custom action collection.

#### Input Parameters

* data – A data object representing the SP.UserCustomAction object properties.

#### Data Properties

Refer to the SP.UserCustomAction REST documentation.

### addExistingContentType

Refer to the SP.ContentTypeCollection addAvailableContentType method.

### addField

Refer to the SP.FieldCollection addField method.

### addFieldAsXml

Refer to the SP.FieldCollection createFieldAsXml method.

### addFile

Refer to the SP.FileCollection add method.

### addList

This method will add a list to the web’s list collection.

#### Input Parameters

* data – A data object representing the SP.List object properties.

#### Data Properties

Refer to the SP.List REST documentation.

### addPermission

This method will add a role definition to the web’s role definition collection.

#### Input Parameters

* data – A data object representing the SP.RoleDefinition object properties.

#### Data Properties

Refer to the SP.RoleDefinition REST documentation.

### addSiteGroup

Refer to the SP.RoleAssignmentCollection addRoleAssignment method.

### addSubFolder

This method adds a sub-folder, by the specified sub-folder name.

#### Input Parameters

* name – A string value, representing the name of the sub-folder.

### addWeb

Refer to the SP.WebCollection add method.

### applyTheme

This method applies the specified theme to the web.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* colorpaletteurl – A string value, representing the server-relative url of the color palette file.
* fontschemeurl – A string value
* backgroundimageurl
* sharegenerated

### applyWebTemplate

This method applies the specified site definition or template to the web site, which has no template applied to it.

#### Input Parameters

* A string value, representing the site definition or web template name.

### breakRoleInheritance

This method creates unique role assignments for the web.

#### Input Parameters

* data – A data object containing the function parameters.

#### Data Properties

* copyroleassignments – A boolean value, representing whether to copy the role assignments from the web.
* clearsubscopes – A boolean value, representing whether to clear the child role assignments.

### deleteObject

This method will delete the web.

### doesPushNotificationSubscriberExist

This method checks whether the push notification subscriber exist for the current user, by the specified device application id.

#### Input Parameters

* A string value, representing the device application id.

### doesUserHavePermissions

This method returns a boolean value, representing whether the user has the specified permissions.

#### Input Parameters

* data – A data object, representing the SP.BasePermissions collection.

#### Data Properties

* High – An int32 value, representing the highest value for the SP.BasePermissions collection.
* Low – An int32 value, representing the lowest value for the SP.BasePermissions collection.

### ensureUser

This method returns a SP.User object, based on the user’s specified login name.

#### Input Parameters

* data – A data object, representing the user’s login name property.

#### Data Properties

* logonName – A string value, representing the user’s login name property.

### executeRemoteLOB

This method sends data to an OData service.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* inputStream – The OData input object, used for create or update operations only.

### getAppBdcCatalog

This method will return a SP.BusinessData.AppBdcCatalog object, for the current application instance.

### getAppBdcCatalogForAppInstance

This method will return a SP.BusinessData.AppBdcCatalog object, based on the specified application instance id.

#### Input Parameters

* A string value, representing the application instance id.

### getAppInstanceById

This method will return a SP.AppInstance object, based on the specified application instance id.

#### Input Parameters

* A string value, representing the application instance id.

### getAppInstancesByProductId

This method will return a SP.AppInstanceCollection object, based on the application’s specified product id.

#### Input Parameters

* A string value, representing the application’s product id.

### getAvailableWebTemplates

This method returns a SP.WebTemplateCollection object, based on the specified parameters.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* lcid – An int64 value, representing the LCID of the site templates to get.
* doincludecrosslanguage – A boolean value, representing whether to include language-neutral site templates.

### getCatalog

This method returns a SP.List object, specified by the gallery type.

#### Input Parameters

* An int32 value, representing the SP.ListTemplateType enumeration value.

### getChanges

This method returns a SP.ChangeCollection object, representing the change log that have occurred within the web, by the specified query.

#### Input Parameters

* data – A data object representing a SP.ChangeQuery object.

#### Data Properties

Refer to the SP.ChangeQuery REST documentation.

### getContentType

This method returns a SP.ContentType object, based on the specified content type name.

#### Input Parameters

* name – A string value, representing the content type’s name property.

### getContentTypeById

This method returns a SP.ContentType object, based on the specified content type id.

#### Input Parameters

* id – A string value, representing the content type’s id property.

### getContextWebInformation

This method returns a SP.ContextWebInformation object.

### getCustomAction

This method returns a SP.UserCustomAction object, based on the specified title.

#### Input Parameters

* title – A string value, representing the user custom action’s title or name property.

### getCustomListTemplates

This method returns a SP.ListTemplateCollection object.

### getDocumentLibraries

This method returns a SP.DocumentLibraryInformation object, based on the specified url.

#### Input Parameters

* A string value, representing the full web url.

### getEntity

This method returns a SP.BusinessData.Entity object, based on the specified parameters.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* namespace – A string value, representing the namespace of the external content type.
* name – A string value, representing the name of the external content type.

### getField

This method returns a SP.Field object, based on specified title.

#### Input Parameters

* title – A string value, representing one of the field’s following properties:
  + Title
  + InternalName
  + StaticName

### getFieldById

This method returns a SP.Field object, based on the specified field id property.

#### Input Parameters

* id – A string value, representing the field’s id property.

### getFieldByInternalName

This method returns a SP.Field object, based on the specified field internal name property.

#### Input Parameters

* name – A string value, representing the field’s internal name property.

### getFieldByStaticName

This method returns a SP.Field object, based on the specified field static name property.

#### Input Parameters

* name – A string value, representing the field’s static name property.

### getFieldByTitle

This method returns a SP.Field object, based on the specified field title property.

#### Input Parameters

* title – A string value, representing the field’s title property.

### getFile

This method returns a SP.File object located in the web’s root folder, based on the specified name.

#### Input Parameters

* name – A string value, representing the name of the file.

### getFileByServerRelativeUrl

This method returns a SP.File object, based on the specified url.

#### Input Parameters

* A string value, representing the file’s server-relative url.

### getFolderByServerRelativeUrl

This method returns a SP.Folder object, based on the specified url.

#### Input Parameters

* A string value, representing the folder’s server-relative url.

### getList

This method returns a SP.List object, based on the specified url.

#### Input Parameters

* A string value, representing the list’s site-relative url.

### getListById

This method returns a SP.List object, based on the specified id.

#### Input Parameters

* A string value, representing the list’s id property.

### getListByTitle

This method returns a SP.List object, based on the specified title.

#### Input Parameters

* A string value, representing the list’s title property.

### getPushNotificationSubscriber

This method returns a SP.PushNotificationSubscriber object, based on the device’s application instance id.

#### Input Parameters

* A string value, representing the device’s application instance id.

### getPushNotificationSubscriberByArgs

This method returns a SP.PushNotificationSubscriberCollection object, based on the specified query.

#### Input Parameters

* A string value, representing the arguments to filter the results.

### getPushNotificationSubscribersByUser

This method returns a SP.PushNotificationSubscriberCollection object, based on the user’s specified login name.

#### Input Parameters

* A string value, representing the user’s login name.

### getSiteGroupById

Refer to the SP.GroupCollection getById method.

### getSiteGroupByName

Refer to the SP.GroupCollection getByName method.

### getSubFolder

This method returns a SP.Folder object, based on the specified by the sub-folder name.

#### Input Parameters

* name – A string value, representing the sub-folder name.

### getSubwebsFilteredForCurrentUser

This method returns a SP.WebInformationCollection object, based on the specified parameters.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* nwebtemplatefilter – An int32 value, representing the template id.
* nconfigurationfilter – An int16 value, representing the site id.

### getUserById

This method returns a SP.User object, based on the user’s specified id.

#### Input Parameters

* An int32 value, representing the user’s id property.

### getUserByLogin

This method returns a SP.User object, based on the user’s specified login name.

#### Input Parameters

* An int32 value, representing the user’s login name property.

### getUserEffectivePermissions

This method returns a SP.BasePermissions object, representing the set of permissions for the specified user.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* user – A string value, representing the login name of the user.

### getWebUrlFromPageUrl

This method returns a string value, representing the site url from the specified page url.

#### Input Parameters

* A string value, representing the full page url.

### hasAccess

This method returns a boolean value, representing whether the specified user has the specified permissions.

#### Input Parameters

* userName – A string value, representing the user’s login name.
* permissions – An array, representing the SP.PermissionKind enumeration types.

### loadAndInstallApp

This method uploads and installs an app package to the site.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* appPackageStream – A stream, representing the app package.

### loadAndInstallAppInSpecifiedLocale

This method uploads and installs an app package, in a specified locale.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* appPackageStream – A stream, representing the app package.
* installationLocaleLCID – An int32 value, representing the locale to create the app instance in.

### loadApp

This method uploads an app package and creates an instance from it.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* appPackageStream – A stream, representing the app package.
* installationLocaleLCID – An int32 value, representing the locale to create the app instance in.

### mapsToIcon

This method returns a string value, representing the name of the image file for the icon that is used to represent the specified file.

#### Input Parameters

* data – A data object representing the function parameters.

#### Data Properties

* filename – A string value, representing the file name.
* progid – A string value, representing the program id of the application that was used to create the file, in the form OLEServerName.ObjectName.
  + Examples: Excel.Sheet or PowerPoint.Slide
* size – An int32 value, representing the size of the icon.
  + 16x16 pixels – 0
  + 32x32 pixels – 1

### processExternalNotification

This method processes a notification from an external system.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* stream – A stream, representing the notification message from the external system.

### registerPushNotificationSubscriber

This method registers the subscriber for push notification over the site.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* deviceappinstanceid – A string value, representing the device’s app instance id.
* servicetoken – A string value, representing the token provided by the notification service to the device to receive notifications.

### resetRoleInheritance

This method resets the role inheritance, and inherits the role assignments for the web.

### sendEmail

This method sends an email, by the specified parameters.

#### Input Parameters

* data – A data object, representing the function parameters.

#### Data Properties

* Body – A string value, representing the body of the email.
* From – A string value, representing who the email is sent from.
* Subject – A string value, representing the subject of the email.
* To – An array object, representing the email addresses to send the email to.
  + Format – { results: [“email@domain.com”] }

### unregisterPushNotifictionSubscriber

This method unregisters the subscriber for push notifications from the web.

#### Input Parameters

* A string value, representing the device’s app instance id.

### update

This method will update the web properties.

#### Input Parameters

* data – A data object containing the SP.Web properties.

## SP.WebCollection

### add

This method adds a web to the collection.

#### Input Parameters

* data – A data object representing the SP.WebCreationInformation object properties.

#### Data Properties

Refer to the SP.WebCreationInformation REST documentation.

# User Profile Service Methods

## SP.Social.SocialRestActor

### createPost

This method creates a root post in the specified site feed.

#### Input Parameters

### getFeed

## SP.Social.SocialRestFeedManager

## SP.Social.SocialRestThread