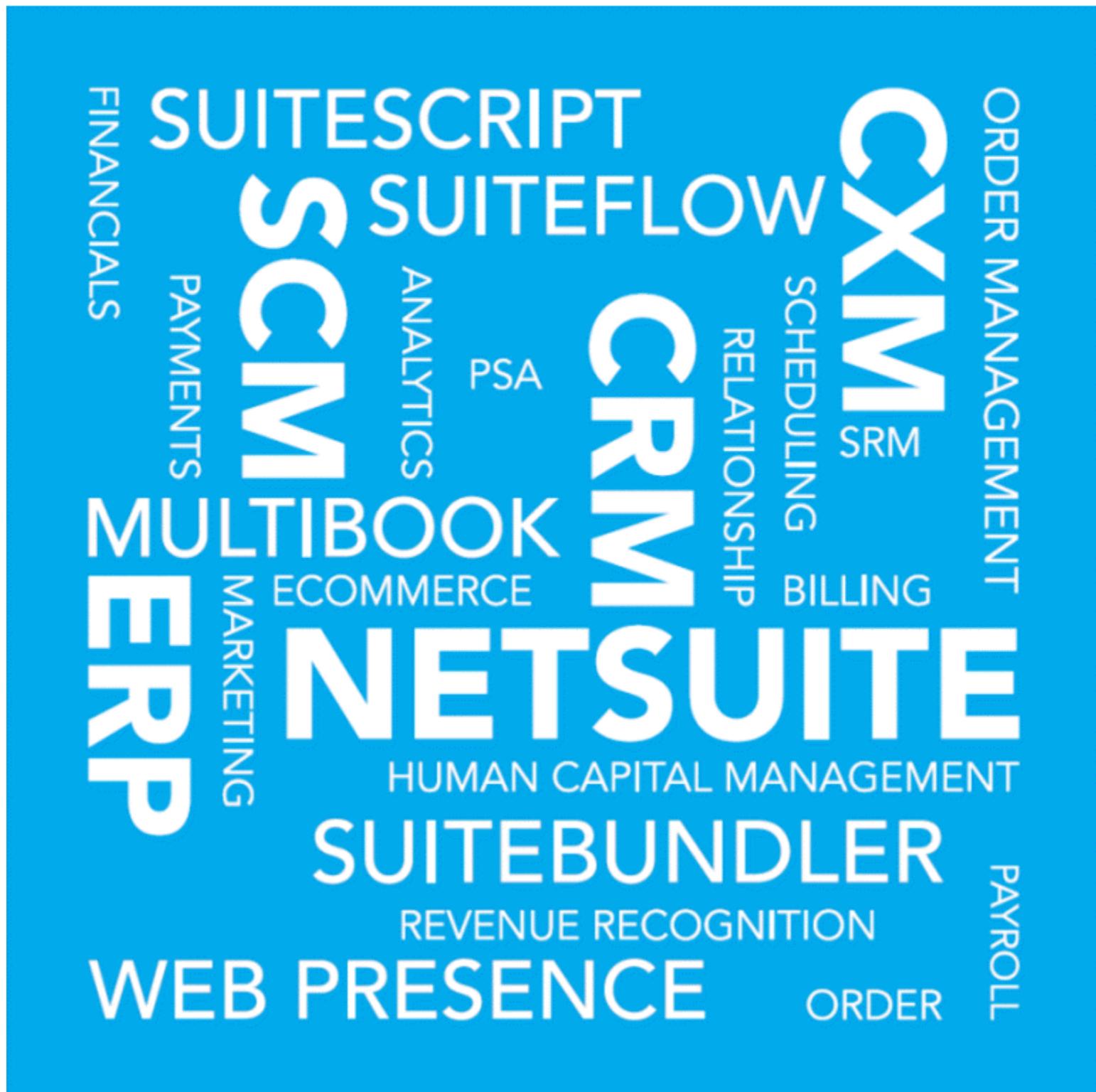


SuiteScript 2.0 API Reference



Copyright © 2005, 2020, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc.

AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

If this document is in public or private pre-General Availability status:

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

If this document is in private pre-General Availability status:

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and

should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Sample Code

Oracle may provide sample code in SuiteAnswers, the Help Center, User Guides, or elsewhere through help links. All such sample code is provided "as is" and "as available", for use only with an authorized NetSuite Service account, and is made available as a SuiteCloud Technology subject to the SuiteCloud Terms of Service at www.netsuite.com/tos.

Oracle may modify or remove sample code at any time without notice.

No Excessive Use of the Service

As the Service is a multi-tenant service offering on shared databases, Customer may not use the Service in excess of limits or thresholds that Oracle considers commercially reasonable for the Service. If Oracle reasonably concludes that a Customer's use is excessive and/or will cause immediate or ongoing performance issues for one or more of Oracle's other customers, Oracle may slow down or throttle Customer's excess use until such time that Customer's use stays within reasonable limits. If Customer's particular usage pattern requires a higher limit or threshold, then the Customer should procure a subscription to the Service that accommodates a higher limit and/or threshold that more effectively aligns with the Customer's actual usage pattern.

Beta Features

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any

operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Send Us Your Feedback

We'd like to hear your feedback on this document.

Answering the following questions will help us improve our help content:

- Did you find the information you needed? If not, what was missing?

- Did you find any errors?
- Is the information clear?
- Are the examples correct?
- Do you need more examples?
- What did you like most about this document?

Click [here](#) to send us your comments. If possible, please provide a page number or section title to identify the content you're describing.

To report software issues, contact NetSuite Customer Support.

Table of Contents

SuiteScript 1.0 to SuiteScript 2.0 API Map	1
SuiteScript 1.0 to SuiteScript 2.0 API Map – Functions (napi)	1
SuiteScript 1.0 to SuiteScript 2.0 API Map – Objects (nlobj)	11
SuiteScript 2.0 Global Objects	35
define Object	35
define(moduleObject)	36
define(id, [dependencies], moduleObject)	37
require Function	40
require([dependencies], callback)	40
require Configuration	42
log Object	43
util Object	44
toString()	44
JSON object	45
JSON.parse(text)	45
JSON.stringify(obj)	46
Promise Object	47
SuiteScript 2.0 Modules	50
N/action Module	53
action.Action	58
action.execute(options)	68
action.execute.promise(options)	70
action.executeBulk(options)	71
action.find(options)	73
action.find.promise(options)	74
action.get(options)	75
action.get.promise(options)	77
N/auth Module	78
auth.changeEmail(options)	79
auth.changePassword(options)	80
N/cache Module	81
cache.Cache	84
cache.getCache(options)	90
cache.Scope	91
N/certificateControl Module	92
certificateControl.Certificate	97
certificateControl.createCertificate(options)	106
certificateControl.findCertificates(options)	107
certificateControl.findUsages(options)	108
certificateControl.deleteCertificate(options)	109
certificateControl.loadCertificate(options)	110
certificateControl.Operation	111
certificateControl.Operator	112
certificateControl.Type	113
N/commerce Modules	114
N/commerce/recordView Module	114
N/compress Module	120
compress.Archiver	122
compress.gzip(options)	125
compress.gunzip(options)	126
compress.createArchiver()	127
compress.Type	128
N/config Module	129

config.load(options)	130
config.Type	131
N/crypto Module	132
crypto.Cipher	137
crypto.CipherPayload	139
crypto.Decipher	141
crypto.Hash	143
crypto.Hmac	145
crypto.SecretKey	147
crypto.createCipher(options)	149
crypto.createDecipher(options)	150
crypto.createHash(options)	151
crypto.createHmac(options)	152
crypto.createSecretKey(options)	152
crypto.EncryptionAlg	153
crypto.HashAlg	154
crypto.Padding	155
N/crypto/certificate Module	156
certificate.SignedXml	159
certificate.Signer	160
certificate.Verifier	163
certificate.createSigner(options)	165
certificate.createVerifier(options)	165
certificate.verifyXmlSignature(options)	166
certificate.signXml(options)	166
certificate.HashAlg	167
N/currency Module	167
currency.exchangeRate(options)	168
N/currentRecord Module	170
currentRecord.Column	178
currentRecord.CurrentRecord	182
currentRecord.Field	226
currentRecord.Sublist	236
currentRecord.get()	239
currentRecord.get.promise()	240
N/email Module	241
email.send(options)	242
email.send.promise(options)	246
email.sendBulk(options)	247
email.sendBulk.promise(options)	251
email.sendCampaignEvent(options)	251
email.sendCampaignEvent.promise(options)	254
N/encode Module	254
encode.convert(options)	255
encode.Encoding	256
N/error Module	257
error.SuiteScriptError	260
error.UserEventError	265
error.create(options)	271
error.Type	272
N/file Module	278
file.File	283
file.create(options)	299
file.delete(options)	301
file.load(options)	302

file.Encoding	303
file.Type	304
file.Reader	305
N/format Module	308
format.format(options)	311
format.parse(options)	312
format.Type	314
format.Timezone	316
N/format/i18n Module	320
format.CurrencyFormatter	324
format.NumberFormatter	327
format.PhoneNumberFormatter	331
format.spellOut(options)	336
format.getCurrencyFormatter(options)	336
format.getNumberFormatter(options)	337
format.NegativeNumberFormat	338
format.Currency	339
N/http Module	339
http.ClientResponse	344
http.ServerRequest	347
http.ServerResponse	355
http.get(options)	368
http.get.promise(options)	369
http.delete(options)	370
http.delete.promise(options)	371
http.post(options)	372
http.post.promise(options)	374
http.put(options)	375
http.put.promise(options)	376
http.request(options)	377
http.request.promise(options)	378
http.CacheDuration	379
http.Method	380
http.RedirectType	381
N/https Module	381
https.SecureString	388
https.createSecureKey(options)	392
https.createSecureKey.promise(options)	393
https.createSecureString(options)	394
https.createSecureString.promise(options)	395
https.ClientResponse	396
https.ServerRequest	398
https.ServerResponse	405
https.get(options)	417
https.get.promise(options)	418
https.delete(options)	419
https.delete.promise(options)	420
https.post(options)	421
https.post.promise(options)	423
https.put(options)	424
https.put.promise(options)	425
https.request(options)	426
https.request.promise(options)	428
https.requestRestlet(options)	429
https.requestSuiteTalkRest(options)	430

https.CacheDuration	432
https.Encoding	433
https.HashAlg	433
https.Method	434
https.RedirectType	435
N/https/clientCertificate Module	436
clientCertificate.post(options)	437
clientCertificate.get(options)	438
clientCertificate.put(options)	438
clientCertificate.delete(options)	439
clientCertificate.request(options)	440
N/keyControl Module	440
keyControl.Key	443
keyControl.createKey(options)	447
keyControl.findKeys(options)	448
keyControl.deleteKey(options)	449
keyControl.loadKey(options)	450
keyControl.Operator	451
N/log Module	452
log.audit(options)	455
log.debug(options)	456
log.emergency(options)	457
log.error(options)	458
N/piremoval Module	459
piremoval.PiRemovalTask	462
piremoval.PiRemovalTaskStatus	472
piremoval.PiRemovalTaskLogItem	474
piremoval.createTask(options)	478
piremoval.deleteTask(options)	480
piremoval.getTaskStatus(options)	481
piremoval.loadTask(options)	482
N/plugin Module	483
plugin.findImplementations(options)	484
plugin.loadImplementation(options)	485
N/portlet Module	486
portlet.resize	488
portlet.refresh	489
N/query Module	489
Scripting with the N/query Module	507
Formulas in the N/query Module	511
Relative Dates in the N/query Module	513
SuiteQL in the N/query Module	515
query.Column	521
query.Component	529
query.Condition	549
query.Page	556
query.PagedData	561
query.PageRange	565
query.Period	568
query.Query	570
query.RelativeDate	603
query.Result	607
query.ResultSet	610
query.Sort	615
query.SuiteQL	621

query.create(options)	627
query.createPeriod(options)	630
query.createRelativeDate(options)	631
query.delete(options)	633
query.listTables(options)	634
query.load(options)	635
query.load.promise(options)	636
query.runSuiteQL(options)	638
query.runSuiteQLPaged(options)	639
query.Aggregate	641
query.DateId	642
query.FieldContext	644
query.Operator	645
query.PeriodAdjustment	648
query.PeriodCode	649
query.PeriodType	651
query.RelativeDateRange	651
query.ReturnType	657
query.SortLocale	658
query.Type	664
N/record Module	674
record.Column	695
record.Field	701
record.Macro	708
record.Record	714
record.Sublist	778
record.attach(options)	783
record.attach.promise(options)	785
record.copy(options)	787
record.copy.promise(options)	789
record.create(options)	791
record.create.promise(options)	793
record.delete(options)	795
record.delete.promise(options)	796
record.detach(options)	798
record.detach.promise(options)	799
record.load(options)	801
record.load.promise(options)	803
record.submitFields(options)	805
record.submitFields.promise(options)	807
record.transform(options)	809
record.transform.promise(options)	813
record.Type	815
N/recordContext Module	818
recordContext.RecordContext	819
recordContext.getContext(options)	819
recordContext.ContextType	821
N/redirect Module	821
redirect.redirect(options)	823
redirect.toRecord(options)	824
redirect.toRecordTransform(options)	825
redirect.toSavedSearch(options)	827
redirect.toSavedSearchResult(options)	827
redirect.toSearch(options)	828
redirect.toSearchResult(options)	829

redirect.toSuitelet(options)	830
redirect.toTaskLink(options)	831
N/render Module	831
render.EmailMergeResult	836
render.TemplateRenderer	837
render.bom(options)	848
render.create()	849
render.mergeEmail(options)	850
render.packingSlip(options)	851
render.pickingTicket(options)	852
render.statement(options)	853
render.transaction(options)	854
render.xmlToPdf(options)	855
render.DataSource	856
render.PrintMode	857
N/runtime Module	858
runtime.Script	862
runtime.Session	868
runtime.User	870
runtime.getCurrentScript()	878
runtime.getCurrentSession()	878
runtime.getCurrentUser()	879
runtime.isFeatureInEffect(options)	880
runtime.accountId	881
runtime.envType	881
runtime.executionContext	882
runtime.processorCount	882
runtime.queueCount	883
runtime.version	884
runtime.ContextType	885
runtime.EnvType	886
runtime.Permission	887
N/search Module	887
search.Search	899
search.Result	912
search.Column	919
search.Filter	927
search.ResultSet	931
search.Page	936
search.PagedData	942
search.PageRange	947
search.Setting	949
search.create(options)	952
search.create.promise(options)	955
search.createSetting(options)	956
search.load(options)	958
search.load.promise(options)	960
search.delete(options)	961
search.delete.promise(options)	962
search.duplicates(options)	963
search.duplicates.promise(options)	964
search.global(options)	965
search.global.promise(options)	966
search.lookupFields(options)	967
search.lookupFields.promise(options)	969

search.createColumn(options)	970
search.createFilter(options)	971
search.Operator	973
search.Sort	974
search.Summary	975
search.Type	976
N/sftp Module	979
Setting up an SFTP Transfer	985
SFTP Authentication	986
Supported Cipher Suites and Host Key Types	988
Supported SuiteScript File Types	988
sftp.Connection	990
sftp.createConnection(options)	1000
sftp.MAX_CONNECT_TIMEOUT	1002
sftp.MIN_CONNECT_TIMEOUT	1003
sftp.MAX_PORT_NUMBER	1003
sftp.MIN_PORT_NUMBER	1004
sftp.DEFAULT_PORT_NUMBER	1004
sftp.Sort	1005
N/sso Module	1006
sso.generateSuiteSignOnToken(options)	1008
N/task Module	1010
task.CsvImportTask	1027
task.CsvImportTaskStatus	1032
task.EntityDeduplicationTask	1034
task.EntityDeduplicationTaskStatus	1039
task.MapReduceScriptTask	1041
task.MapReduceScriptTaskStatus	1045
task.QueryTask	1056
task.QueryTaskStatus	1064
task.RecordActionTask	1069
task.RecordActionTaskStatus	1074
task.ScheduledScriptTask	1081
task.ScheduledScriptTaskStatus	1085
task.SearchTask	1088
task.SearchTaskStatus	1095
task.SuiteQLTask	1099
task.SuiteQLTaskStatus	1108
task.WorkflowTriggerTask	1113
task.WorkflowTriggerTaskStatus	1117
task.checkStatus(options)	1118
task.create(options)	1119
task.ActionCondition	1125
task.DedupeEntityType	1125
task.DedupeMode	1126
task.MapReduceStage	1127
task.MasterSelectionMode	1128
task.TaskStatus	1129
task.TaskType	1130
N/task/accounting/recognition Module	1131
recognition.MergeArrangementsTask	1137
recognition.MergeArrangementsTaskStatus	1143
recognition.MergeElementsTask	1147
recognition.create(options)	1151
recognition.checkStatus(options)	1152

recognition.TaskStatus	1153
recognition.TaskType	1154
N/transaction Module	1154
transaction.void(options)	1156
transaction.void.promise(options)	1157
transaction.Type	1158
N/translation Module	1161
translation.Handle	1166
translation.Translator	1167
translation.get(options)	1169
translation.load(options)	1170
translation.selectLocale(options)	1172
translation.Locale	1173
N/ui Modules	1176
N/ui/dialog Module	1176
N/ui/message Module	1184
N/ui/serverWidget Module	1190
N/url Module	1325
url.format(options)	1328
url.resolveDomain(options)	1329
url.resolveRecord(options)	1330
url.resolveScript(options)	1331
url.resolveTaskLink(options)	1332
url.HostType	1333
N/util Module	1334
util.isArray(obj)	1336
util.isBoolean(obj)	1337
util.isDate(obj)	1337
util.isAsyncFunction(obj)	1338
util.isFunction(obj)	1339
util.isNumber(obj)	1340
utilisObject(obj)	1340
util.isRegExp(obj)	1341
util.isString(obj)	1342
util.each(iterable, callback)	1343
util.extend(receiver, contributor)	1343
N/workflow Module	1345
workflow.initiate(options)	1346
workflow.trigger(options)	1347
N/xml Module	1348
xml.Parser	1357
xml.XPath	1359
xml.Node	1360
xml.Document	1379
xml.Element	1395
xml.Attr	1409
xml.escape(options)	1411
xml.validate(options)	1412
xml.NodeType	1413

SuiteScript 1.0 to SuiteScript 2.0 API Map



Important: These topics are a work in progress. Some items are currently missing or do not have content. Additional updates are forthcoming.

These topics map SuiteScript 1.0 APIs to their corresponding SuiteScript 2.0 APIs. Keep the following in mind when using these mappings:

- Some SuiteScript 1.0 APIs do not have a SuiteScript 2.0 equivalent.
- There is not always a one to one mapping between SuiteScript 1.0 and SuiteScript 2.0. Each SuiteScript 1.0 API is listed only one time, but it may map to several SuiteScript 2.0 APIs.
- These mappings **do not include** SuiteScript 1.0 deprecated APIs.
- These mappings **do not include** new SuiteScript 2.0 functionality. To find new SuiteScript 2.0 functionality, go to [SuiteScript 2.0 Modules](#). The table includes a description of, and link to, each module.



Important: If you are using SuiteScript 1.0 for your scripts, consider converting these scripts to SuiteScript 2.0. Use SuiteScript 2.0 to take advantage of new features, APIs, and functionality enhancements. For more information, see the help topic [SuiteScript 2.0 Advantages](#).

These topics group SuiteScript 1.0 APIs into functions (prefixed with "nlapi") and objects (prefixed with "nlobj"). All functions are listed alphabetically in one table. Whereas objects and their members are grouped alphabetically by object name. Each object has its own table containing all object members.

- [SuiteScript 1.0 to SuiteScript 2.0 API Map – Functions \(nlapi\)](#)
- [SuiteScript 1.0 to SuiteScript 2.0 API Map – Objects \(nlobj\)](#)

SuiteScript 1.0 to SuiteScript 2.0 API Map – Functions (nlapi)

This topic maps SuiteScript 1.0 Functions (prefixed with "nlapi") to their corresponding SuiteScript 2.0 APIs. All functions are listed alphabetically in one table.



Note: NetSuite does not support calling SuiteScript 1.0 APIs from SuiteScript 2.0 scripts.



Note: To view a mapping of SuiteScript 1.0 Objects (prefixed with "nlobj") to their corresponding SuiteScript 2.0 APIs, see [SuiteScript 1.0 to SuiteScript 2.0 API Map – Objects \(nlobj\)](#).

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlapiAddDays(d, days)	See Notes	See Notes	<p>This API does not have a SuiteScript 2.0 equivalent.</p> <p>Use the following JavaScript to add or subtract days from a Date object: dateObj.setDate(dateObj.getDate() + or - days)</p> <p>For example:</p> <pre> 1 var tomorrow = new Date(); 2 tomorrow.setDate(tomorrow.getDate() + 1); </pre>

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
			<p>Note: SuiteScript 2.0 is also compatible with third party JavaScript APIs that provide this functionality (for example, Moment.js). For information on using third party APIs with SuiteScript 2.0, see the help topic SuiteScript 2.0 Custom Modules.</p>
nlapiAddMonths(d, months)	See Notes	See Notes	<p>This API does not have a SuiteScript 2.0 equivalent.</p> <p>Use the following JavaScript to add or subtract months from a Date object: dateObj.setMonth(dateObj.getMonth() + or - months)</p> <p>For example:</p> <pre> 1 var today = new Date(); 2 var oneMonthAgo = today.setMonth(today.getMonth() - 1); </pre> <p>Note: SuiteScript 2.0 is also compatible with third party JavaScript APIs that provide this functionality (for example, Moment.js). For information on using third party APIs with SuiteScript 2.0, see the help topic SuiteScript 2.0 Custom Modules.</p>
nlapiAttachRecord(type, id, type2, id2, attributes)	<code>record.attach(options)</code>	N/record Module	<pre> 1 var recordId = record.attach({ 2 record: { 3 type: record.Type.FILE, 4 id: '447' 5 }, 6 to: { 7 type: record.type.CUSTOMER, 8 id: 530 9 } 10 }); </pre>
nlapiCancelLineItem(type)	<code>Record.cancelLine(options)</code> <code>CurrentRecord.cancelLine(options)</code>	N/record Module N/currentRecord Module	-
nlapiCommitLineItem(type)	<code>Record.commitLine(options)</code> <code>CurrentRecord.commitLine(options)</code>	N/record Module N/currentRecord Module	<p>For N/record script samples, see:</p> <ul style="list-style-type: none"> ▪ N/record Module Script Samples ▪ Example: Creating an Inventory Detail Sublist Subrecord
nlapiCopyRecord(type, id, initializeValues)	<code>record.copy(options)</code>	N/record Module	<pre> 1 var recObj = record.copy({ 2 type: record.Type.SALES_ORDER, 3 id: 284, 4 isDynamic: true, 5 defaultValues: { 6 entity: 547 7 } 8 }); var recordId = recObj.save(); </pre>
nlapiCreateAssistant(title, hideHeader)	<code>serverWidget.createAssistant(options)</code>	N/ui/serverWidget Module	-
nlapiCreateCSVImport()	<code>task.create(options)</code>	N/task Module	For script samples, see N/task Module .
nlapiCreateCurrentLineItemSubrecord(sublist, fldname)	<code>Record.getCurrentSublistSubrecord(options)</code> <code>CurrentRecord.getCurrentSublistSubrecord(options)</code>	N/record Module N/currentRecord Module	<p>Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords.</p>

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlapiCreateEmailMerger(templateId)	render.mergeEmail(options)	N/render Module	-
nlapiCreateError(code, details, suppressNotification)	error.create(options)	N/error Module	For a script sample, see N/error Module Script Samples .
nlapiCreateFile(name, type, contents)	file.create(options)	N/file Module	For a script sample, see N/file Module Script Samples .
nlapiCreateForm(title, hideNavbar)	serverWidget.createForm(options)	N/ui/serverWidget Module	For a script sample, see N/ui/serverWidget Module Script Samples
nlapiCreateList(title, hideNavbar)	serverWidget.createList(options)	N/ui/serverWidget Module	-
nlapiCreateRecord(type, initializeValues)	record.create(options)	N/record Module	-
nlapiCreateSearch(type, filters, columns)	search.create(options)	N/search Module	For a script sample, see N/search Module Script Samples .
nlapiCreateSubrecord(fdname)	Record.getSubrecord(options) CurrentRecord.getSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlapiCreateTemplateRenderer()	render.create()	N/render Module	For a script sample, see N/render Module Script Sample .
nlapiDateToString(d, format)	format.format(options)	N/format Module	For a script sample, see N/format Module Script Samples
nlapiDeleteFile(id)	file.delete(options)	N/file Module	-
nlapiDeleteRecord(type, id, initializeValues)	record.delete(options)	N/record Module	-
nlapiDetachRecord(type, id, type2, id2, attributes)	record.detach(options)	N/record Module	-
nlapiDisableField(fdnam, val)	Field.isDisabled	N/currentRecord Module	Note that isDisabled is a property.
nlapiDisableLineItemField(type, fdnam, val)	Field.isDisabled	N/currentRecord Module	Note that isDisabled is a property.
nlapiEditCurrentLineItemSubrecord(sublist, fdname)	Record.getCurrentSublistSubrecord(options) CurrentRecord.getCurrentSublistSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlapiEditSubrecord(fdname)	Record.getSubrecord(options) CurrentRecord.getSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlapiEncrypt(s, algorithm, key)	See Notes	See Notes	For SuiteScript 2.0 encryption, hashing, and HMAC functionality, see the N/crypto Module module. For SuiteScript 2.0 encoding functionality, see the N/encode Module module.
nlapiEscapeXML(text)	xml.escape(options)	N/xml Module	-
nlapiExchangeRate(sourceCurrency, targetCurrency, effectiveDate)	currency.exchangeRate(options)	N/currency Module	For a script sample, see N/currency Module Script Samples .
nlapiFindLineItemMatrixValue(type, fdnam, val, column)	Record.findMatrixSublistLineWithValue(options)	N/record Module N/currentRecord Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
	CurrentRecord.find MatrixSublistLineWith Value(options)		
nlapiFindLineItemValue(type, fldnam, val)	Record.findSublistLine WithValue(options) CurrentRecord.find SublistLineWithValue(options)	N/record Module N/currentRecord Module	-
nlapiFormatCurrency(str)	format.format(options)	N/format Module	Note that SuiteScript 2.0 currency formatting is handled by the N/format module and not the N/currency module. For a script sample, see N/format Module Script Samples .
nlapiGetContext()	runtime.getCurrent Script() runtime.getCurrent Session() runtime. getCurrentUser()	N/runtime Module	For a script sample, see N/runtime Module Script Samples .
nlapiGetCurrentLineItemDate TimeValue(type, fieldId, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlapiGetCurrentLineItemIndex(type)	Record.getCurrent SublistIndex(options) CurrentRecord.get CurrentSublistIndex(options)	N/record Module N/currentRecord Module	-
nlapiGetCurrentLineItemMatrix Value(type, fldnam, column)	CurrentRecord.get CurrentMatrixSublist Value(options) Record.getCurrent MatrixSublistValue(options)	N/record Module N/currentRecord Module	-
nlapiGetCurrentLineItemText(type, fldnam)	Record.getCurrent SublistText(options) CurrentRecord.get CurrentSublistText(options)	N/record Module N/currentRecord Module	-
nlapiGetCurrentLineItemValue(type, fldnam)	Record.getCurrent SublistValue(options) CurrentRecord.get CurrentSublistValue(options)	N/record Module N/currentRecord Module	-
nlapiGetCurrentLineItemValues(type, fldnam)	Record.getCurrent SublistValue(options) CurrentRecord.get CurrentSublistValue(options)	N/record Module N/currentRecord Module	-
nlapiGetDateTimeValue(fieldId, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlapiGetDepartment()	User.department	N/runtime Module	-
nlapiGetField(fldnam)	Record.getField (options) CurrentRecord.getField (options)	N/record Module N/currentRecord Module	-
nlapiGetFieldText(fldnam)	Record.getText (options)	N/record Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
	CurrentRecord.getText(options)	N/currentRecord Module	
nlapiGetFieldTexts(fldnam)	Record.getText(options) CurrentRecord.getText(options)	N/record Module N/currentRecord Module	-
nlapiGetFieldValue(fldnam)	Record.getValue(options) CurrentRecord.getValue(options)	N/record Module N/currentRecord Module	-
nlapiGetFieldValues(fldnam)	Record.getValue(options) CurrentRecord.getValue(options)	N/record Module N/currentRecord Module	-
nlapiGetJobManager(jobType)	task.create(options)	N/task Module	For a script sample, see N/task Module .
nlapiGetLineItemCount(type)	Record.getLineCount(options) CurrentRecord.getLineCount(options)	N/record Module N/currentRecord Module	-
nlapiGetLineItemDateTimeValue(type, fieldId, lineNum, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlapiGetLineItemField(type, fldnam, linenum)	Record.getSublistField(options) CurrentRecord.getSublistField(options)	N/record Module N/currentRecord Module	SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiGetLineItemMatrixField(type, fldnam, linenum, column)	Record.getMatrixSublistField(options) CurrentRecord.getMatrixSublistField(options)	N/record Module N/currentRecord Module	SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiGetLineItemMatrixValue(type, fldnam, linenum, column)	Record.getMatrixSublistValue(options) CurrentRecord.getMatrixSublistValue(options)	N/record Module N/currentRecord Module	SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiGetLineItemText(type, fldnam, linenum)	Record.getSublistText(options) CurrentRecord.getSublistText(options)	N/record Module N/currentRecord Module	SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiGetLineItemValue(type, fldnam, linenum)	Record.getSublistValue(options) CurrentRecord.getSublistValue(options)	N/record Module N/currentRecord Module	SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiGetLineItemValues(type, fldname, linenum)	Record.getSublistValue(options) CurrentRecord.getSublistValue(options)	N/record Module N/currentRecord Module	Method returns an array for multi-select fields. SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiGetLocation()	User.location	N/runtime Module	Note that location is a property.
nlapiGetLogin()	auth.changeEmail(options)	N/auth Module	For a script sample, see N/auth Module Script Sample .

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
	auth.changePassword(options)		
nlapiGetMatrixCount(type, fldnam)	Record.getMatrixHeaderCount(options) CurrentRecord.getMatrixHeaderCount(options)	N/record Module N/currentRecord Module	-
nlapiGetMatrixField(type, fldnam, column)	Record.getMatrixHeaderField(options) CurrentRecord.getMatrixHeaderField(options)	N/record Module N/currentRecord Module	-
nlapiGetMatrixValue(type, fldnam, column)	Record.getMatrixHeaderValue(options) CurrentRecord.getMatrixHeaderValue(options)	N/record Module N/currentRecord Module	-
nlapiGetNewRecord()	See Notes	See Notes	To mimic this functionality in SuiteScript 2.0, use the following code in a beforeLoad(scriptContext) , beforeSubmit(scriptContext) , or afterSubmit(scriptContext) user event script. <pre>1 function afterSubmit(context) { 2 var newRec = context.newRecord; 3 }</pre> <p>For additional information and a full script sample, see the help topic SuiteScript 2.0 User Event Script Type</p>
nlapiGetOldRecord()	See Notes	See Notes	To mimic this functionality in SuiteScript 2.0, use the following code in a beforeSubmit(scriptContext) or afterSubmit(scriptContext) user event script. <pre>1 function afterSubmit(context) { 2 var oldRec = context.oldRecord; 3 }</pre> <p>For additional information and a full script sample, see the help topic SuiteScript 2.0 User Event Script Type</p>
nlapiGetRecordId()	Record.id CurrentRecord.id	N/record Module N/currentRecord Module	-
nlapiGetRecordType()	Record.type CurrentRecord.type	N/record Module N/currentRecord Module	To get the current record type in a client script, use CurrentRecord.type: <pre>1 function saveRec(context) { 2 var rec = context.currentRecord; 3 var recType = rec.type; 4 }</pre> <p>To get the current record type in a server-side script, use Record.type in a beforeLoad(scriptContext), beforeSubmit(scriptContext), or afterSubmit(scriptContext) user event script: <pre>1 function beforeSubmit(context) { 2 var newRec = context.newRecord; 3 var recType = newRec.type; 4 }</pre> </p>
nlapiGetRole()	User.role	N/runtime Module	-
nlapiGetSubsidiary()	User.subsidiary	N/runtime Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlapi GetUser()	runtime.getCurrentUser()	N/runtime Module	-
nlapiInitiateWorkflow(recordtype, id, workflowid, initialvalues)	workflow.initiate(options)	N/workflow Module	For a script sample, see N/workflow Module Script Sample .
nlapiInitiateWorkflowAsync(recordType, id, workflowId, initialValues)	task.WorkflowTrigger Task	N/task Module	-
nlapiInsertLineItem(type, line)	Record.insertLine(options) CurrentRecord.insertLine(options)	N/record Module N/currentRecord Module	-
nlapiInsertLineItemOption(type, fldnam, value, text, selected)	Field.insertSelectOption(options)	N/currentRecord Module	-
nlapiInsertSelectOption(fldnam, value, text, selected)	Field.insertSelectOption(options)	N/currentRecord Module	-
nlapiIsLineItemChanged(type)	Sublist.isChanged	N/record Module	Note that isChanged is a property 1 record.getSublist("addressbook").isChanged
nlapiLoadConfiguration(type)	config.load(options)	N/config Module	For a script sample, see N/config Module Script Sample .
nlapiLoadFile(id)	file.load(options)	N/file Module	For a script sample, see N/file Module Script Samples .
nlapiLoadRecord(type, id, initializeValues)	record.load(options)	N/record Module	-
nlapiLoadSearch(type, id)	search.load(options)	N/search Module	For a script sample, see N/search Module Script Samples .
nlapiLogExecution(type, title, details)	log.audit(options) log.debug(options) log.emergency(options) log.error(options)	N/log Module	For a script sample, see N/log Module Script Sample .
nlapiLookupField(type, id, fields, text)	search.lookupFields(options)	N/search Module	-
nlapiOutboundSSO(id)	sso.generateSuiteSignOnToken(options)		For a script sample, see N/sso Module Script Sample .
nlapiPrintRecord(type, id, mode, properties)	render.bom(options) render.packingSlip(options) render.pickingTicket(options) render.statement(options) render.transaction(options)	N/render Module	For a script sample, see N/render Module Script Sample .
nlapiRefreshLineItems(type)	-	-	This API does not have a SuiteScript 2.0 equivalent.
nlapiRefreshPortlet()	portlet.refresh	N/portlet Module	For a script sample, see N/portlet Module Script Sample
nlapiRemoveCurrentLineItemSubrecord(sublist, fldname)	Record.removeCurrentSublistSubrecord(options) CurrentRecord.removeCurrentSublistSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlapiRemoveLineItem(type, line)	Record.removeLine(options) CurrentRecord.removeLine(options)	N/record Module N/currentRecord Module	-
nlapiRemoveLineItemOption(type, fldnam, value)	Field.removeSelectOption(options)	N/currentRecord Module	-
nlapiRemoveSelectOption(fldnam, value)	Field.removeSelectOption(options)	N/currentRecord Module	-
nlapiRemoveSubrecord(fldname)	Record.removeSubrecord(options) CurrentRecord.removeSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlapiRequestURL(url, postdata, headers, callback, httpMethod)	http.delete(options) http.get(options) http.post(options) http.put(options) http.request(options)	N/http Module	-
nlapiRequestURLWithCredentials(credentials, url, postdata, headers, httpsMethod)	https.request(options)	N/https Module	Server-side scripts only
nlapiResizePortlet()	portlet.resize	N/portlet Module	For a script sample, see N/portlet Module Script Sample
nlapiResolveURL(type, identifier, id, displayMode)	url.resolveRecord(options) url.resolveScript(options) url.resolveTaskLink(options)	N/url Module	For a script sample, see N/url Module Script Samples .
nlapiScheduleScript(scriptId, deployId, params)	task.create(options)	N/task Module	<pre> 1 var scheduleScriptTaskObj = task.create({ 2 taskType: task.TaskType.SCHEDULED_SCRIPT, 3 //Other Params 4 }); </pre>
nlapiSearchDuplicate(type, fields, id)	search.duplicates(options)	N/search Module	-
nlapiSearchGlobal(keywords)	search.global(options)	N/search Module	-
nlapiSearchRecord(type, id, filters, columns)	search.create(options) search.load(options)	N/search Module	For a script sample, see N/search Module Script Samples .
nlapiSelectLineItem(type, linenum)	Record.selectLine(options) CurrentRecord.selectLine(options)	N/record Module N/currentRecord Module	SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiSelectNewLineItem(type)	Record.select.NewLine(options) CurrentRecord.select.NewLine(options)	N/record Module N/currentRecord Module	-
nlapiSelectNode(node, xpath)	XPath.select(options)	N/xml Module	-
nlapiSelectNodes(node, xpath)	XPath.select(options)	N/xml Module	-
nlapiSelectValue(node, xpath)	See Notes	N/xml Module	To mimic this functionality in SuiteScript 2.0, select a node with XPath.select(options) and then inspect the Node.textContent property.

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlapiSelectValues(node, path)	See Notes	N/xml Module	To mimic this functionality in SuiteScript 2.0, select an array of nodes with XPath.select(options) and then loop through each node's Node.textContent property.
nlapiSendCampaignEmail(campaigneventid, recipientid)	email.sendCampaignEvent(options)	N/email Module	-
nlapiSendEmail(author, recipient, subject, body, cc, bcc, records, attachments, notifySenderOnBounce, internalOnly, replyTo)	email.send(options) email.sendBulk(options)	N/email Module	For a script sample, see N/email Module Script Sample .
nlapiSendFax(author, recipient, subject, body, records, attachments)	N/A	-	This API does not have a SuiteScript 2.0 equivalent.
nlapiSetCurrentLineItemDate TimeValue(type, fieldId, dateTime, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlapiSetCurrentLineItemMatrix Value(type, fldnam, column, value, firefieldchanged, synchronous)	Record.setCurrentMatrixSublistValue(options) CurrentRecord.setMatrixSublistValue(options)	N/record Module N/currentRecord Module	-
nlapiSetCurrentLineItemText(type, fldnam, text, firefieldchanged, synchronous)	Record.setCurrentSublistText(options) CurrentRecord.setSublistText(options)	N/record Module N/currentRecord Module	-
nlapiSetCurrentLineItemValue(type, fldnam, value, firefieldchanged, synchronous)	Record.setCurrentSublistValue(options) CurrentRecord.setSublistValue(options)	N/record Module N/currentRecord Module	-
nlapiSetCurrentLineItemValues(type, fldnam, values, firefieldchanged, synchronous)	Record.setCurrentSublistValue(options) CurrentRecord.setSublistValue(options)	N/record Module N/currentRecord Module	-
nlapiSetDateTimeValue(fieldId, dateTime, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlapiSetText(fldname, txt, firefieldchanged, synchronous)	Record.setText(options) CurrentRecord.setText(options)	N/record Module N/currentRecord Module	-
nlapiSetFieldTexts (fldname, txts, firefieldchanged, synchronous)	Record.setText(options) CurrentRecord.setText(options)	N/record Module N/currentRecord Module	-
nlapiSetFieldValue(fldnam, value, firefieldchanged, synchronous)	Record.setValue(options) CurrentRecord.setValue(options)	N/record Module N/currentRecord Module	-
nlapiSetFieldValues (fldnam, value, firefieldchanged, synchronous)	Record.setValue(options) CurrentRecord.setValue(options)	N/record Module N/currentRecord Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlapiSetLineItemDateTimeValue(type, fieldId, lineNum, dateTime, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlapiSetLineItemValue(type, fldnam, linenum, value)	Record.setSublistValue(options)	N/record Module	SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiSetMatrixValue(type, fldnam, column, value, firefieldchanged, synchronous)	Record.setMatrixHeaderValue(options) CurrentRecord.setMatrixHeaderValue(options)	N/record Module N/currentRecord Module	-
nlapiSetRecoveryPoint()	See Notes	See Notes	The SuiteScript 2.0 Map/Reduce Script Type automatically incorporates yielding.
nlapiSetRedirectURL(type, identifier, id, editmode, parameters)	redirect.redirect(options) redirect.toRecord(options) redirect.toSuitelet(options) redirect.toTaskLink(options)	N/redirect Module	For a script sample, see N/redirect Module Script Samples .
nlapiStringToDate(str, format)	format.parse(options)	N/format Module	For a script sample, see N/format Module Script Samples .
nlapiStringToXML(text)	Parser.fromString(options)	N/xml Module	-
nlapiSubmitConfiguration(name)	Record.save(options)	N/record Module	-
nlapiSubmitCSVImport(nlobjCSVImport)		N/task Module	-
nlapiSubmitRecord(record, doSourcing, ignoreMandatoryFields)	Record.save(options)	N/record Module	-
nlapiSubmitField(type, id, fields, values, doSourcing)	record.submitFields(options)	N/record Module	-
nlapiSubmitFile(file)	File.save()	N/file Module	For a script sample, see N/file Module Script Samples .
nlapiTransformRecord(type, id, transformType, transformValues)	record.transform(options)	N/record Module	-
nlapiTriggerWorkflow(recordtype, id, workflowid, actionid, stateid)	workflow.trigger(options)	N/workflow Module	-
nlapiValidateXML(xmlDocument, schemaDocument, schemaFolderId)	xml.validate(options)	N/xml Module	-
nlapiViewCurrentLineItemSubrecord(sublist, fldname)	CurrentRecord.getCurrentSublistSubrecord(options) Record.getCurrentSublistSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlapiViewLineItemSubrecord(sublist, fldname, linenum)	Record.getSublistSubrecord(options)	N/record Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords . SuiteScript 2.0 begins sublist numbering with 0. SuiteScript 1.0 begins sublist numbering with 1.
nlapiViewSubrecord(fldname)	Record.getSubrecord(options)	N/record Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
	CurrentRecord. getSubrecord(options)	N/currentRecord Module	SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlapiVoidTransaction(transactionType, recordId)	transaction. void(options)	N/transaction Module	For a script sample, see N/transaction Module Script Samples
nlapiXMLToPDF(xmlstring)	render.xmlToPdf (options) TemplateRenderer. renderAsPdf()	N/render Module	Note that TemplateRenderer.renderAsPdf() is equivalent to nlapiXMLToPDF(nlobjEmailMerger.renderToString()). For a script sample, see N/render Module Script Sample .
nlapiXMLToString(xml)	Parser.toString (options)	N/xml Module	-
nlapiYieldScript()	See Notes	See Notes	Note that the SuiteScript 2.0 Map/Reduce Script Type automatically incorporates yielding.

SuiteScript 1.0 to SuiteScript 2.0 API Map – Objects (nlobj)

This topic maps SuiteScript 1.0 Objects (prefixed with “nlobj”) to their corresponding SuiteScript 2.0 APIs. Objects and their members are grouped alphabetically by object name. Each object has its own table containing all object members.



Note: NetSuite does not support calling SuiteScript 1.0 APIs from SuiteScript 2.0 scripts.



Note: To view a mapping of SuiteScript 1.0 Functions (prefixed with “nlapi”) to their corresponding SuiteScript 2.0 APIs, see [SuiteScript 1.0 to SuiteScript 2.0 API Map – Functions \(nlapi\)](#).

- [nlobjAssistant](#)
- [nlobjAssistantStep](#)
- [nlobjButton](#)
- [nlobjColumn](#)
- [nlobjConfiguration](#)
- [nlobjContext](#)
- [nlobjCredentialBuilder](#)
- [nlobjCSVImport](#)
- [nlobjDuplicateJobRequest](#)
- [nlobjEmailMerger](#)
- [nlobjError](#)
- [nlobjField](#)
- [nlobjFieldGroup](#)
- [nlobjFile](#)
- [nlobjForm](#)
- [nlobjFuture](#)
- [nlobjJobManager](#)
- [nlobjList](#)
- [nlobjLogin](#)
- [nlobjMergeResult](#)

- [nlobjPortlet](#)
- [nlobjRecord](#)
- [nlobjRequest](#)
- [nlobjResponse](#)
- [nlobjSearch](#)
- [nlobjSearchColumn](#)
- [nlobjSearchFilter](#)
- [nlobjSearchResult](#)
- [nlobjSearchResultSet](#)
- [nlobjSelectOption](#)
- [nlobjSublist](#)
- [nlobjSubrecord](#)
- [nlobjTab](#)
- [nlobjTemplateRenderer](#)

nlobjAssistant

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjAssistant	serverWidget.Assistant	N/ui/serverWidget Module	-
nlobjAssistant.addField(name, type, label, source, group)	Assistant.addField(options)	N/ui/serverWidget Module	-
nlobjAssistant.addFieldGroup(name, label)	Assistant.addFieldGroup(options)	N/ui/serverWidget Module	-
nlobjAssistant.addStep(name, label)	Assistant.addStep(options)	N/ui/serverWidget Module	-
nlobjAssistant.addSubList(name, type, label)	Assistant.addSublist(options)	N/ui/serverWidget Module	-
nlobjAssistant.getAllFields()	Assistant.getFieldIds()	N/ui/serverWidget Module	-
nlobjAssistant.getAllFieldGroups()	Assistant.getFieldGroupIds()	N/ui/serverWidget Module	-
nlobjAssistant.getAllSteps()	Assistant.getSteps()	N/ui/serverWidget Module	-
nlobjAssistant.getAllSubLists()	Assistant.getSublistIds()	N/ui/serverWidget Module	-
nlobjAssistant.getCurrentStep()	Assistant.currentStep	N/ui/serverWidget Module	Note that currentStep is a property.
nlobjAssistant.getField(name)	Assistant.getField(options)	N/ui/serverWidget Module	-
nlobjAssistant.getFieldGroup(name)	Assistant.getFieldGroup(options)	N/ui/serverWidget Module	-
nlobjAssistant.getLastAction()	Assistant.getLastAction()	N/ui/serverWidget Module	-
nlobjAssistant.getLastStep()	Assistant.getLastStep()	N/ui/serverWidget Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjAssistant.getNextStep()	Assistant.getNextStep()	N/ui/serverWidget Module	-
nlobjAssistant.getStep(name)	Assistant.getStep(options)	N/ui/serverWidget Module	-
nlobjAssistant.getSubList(name)	Assistant.getSublist(options)	N/ui/serverWidget Module	-
nlobjAssistant.hasError()	Assistant.hasErrorHtml()	N/ui/serverWidget Module	-
nlobjAssistant.isFinished()	Assistant.isFinished()	N/ui/serverWidget Module	-
nlobjAssistant.sendRedirect(response)	Assistant.sendRedirect(options)	N/ui/serverWidget Module	-
nlobjAssistant.setCurrentStep(step)	Assistant.currentStep	N/ui/serverWidget Module	Note that currentStep is a property.
nlobjAssistant.setError(html)	Assistant.errorHtml	N/ui/serverWidget Module	Note that errorHtml is a property.
nlobjAssistant.setFieldValues(values)	Assistant.updateDefaultValues(values)	N/ui/serverWidget Module	-
nlobjAssistant.setFinished(html)	Assistant.finishedHtml	N/ui/serverWidget Module	Note that finishedHtml is a property.
nlobjAssistant.setNumbered(hasStepNumber)	Assistant.hideStepNumber	N/ui/serverWidget Module	Note that hideStepNumber is a property.
nlobjAssistant.setOrdered(ordered)	Assistant.isNotOrdered	N/ui/serverWidget Module	Note that isNotOrdered is a property.
nlobjAssistant.setScript(script)	Assistant.clientScriptFileDialog Assistant.clientScriptModulePath	N/ui/serverWidget Module	Note that clientScriptFileDialog and clientScriptModulePath are properties. Use one of these SuiteScript 2.0 properties to attach an ad hoc client script to an assistant.
nlobjAssistant.setShortcut(show)	Assistant.hideAddToShortcutsLink	N/ui/serverWidget Module	Note that hideAddToShortcutsLink is a property.
nlobjAssistant.setSplash(title, text1, text2)	Assistant.setSplash(options)	N/ui/serverWidget Module	-
nlobjAssistant.setTitle(title)	Assistant.title	N/ui/serverWidget Module	Note that title is a property.

nlobjAssistantStep

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjAssistantStep	serverWidget.AssistantStep	N/ui/serverWidget Module	-
nlobjAssistantStep.getAllFields()	AssistantStep.getFieldIds()	N/ui/serverWidget Module	-
nlobjAssistantStep.getAllLineItemFields(group)	AssistantStep.getSublistFieldIds(options)	N/ui/serverWidget Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjAssistantStep.getAllLineItems()	AssistantStep.getSubmittedSublistIds()	N/ui/serverWidget Module	-
nlobjAssistantStep.getFieldValue(name)	AssistantStep.getValue(options)	N/ui/serverWidget Module	-
nlobjAssistantStep.getFieldValues(name)	AssistantStep.getValue(options)	N/ui/serverWidget Module	-
nlobjAssistantStep.getLineItemCount(group)	AssistantStep.getLineCount(options)	N/ui/serverWidget Module	-
nlobjAssistantStep.getLineItemValue(group, name, line)	AssistantStep.getSublistValue(options)	N/ui/serverWidget Module	-
nlobjAssistantStep.getStepNumber()	AssistantStep.stepNumber	N/ui/serverWidget Module	Note that stepNumber is a property.
nlobjAssistantStep.setHelpText(help)	AssistantStep.helpText	N/ui/serverWidget Module	Note that helpText is a property.
nlobjAssistantStep.setLabel(label)	AssistantStep.label	N/ui/serverWidget Module	Note that label is a property.

nlobjButton

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjButton	serverWidget.Button	N/ui/serverWidget Module	-
nlobjButton.setDisabled(disabled)	Button.isDisabled	N/ui/serverWidget Module	Note that isDisabled is a property.
nlobjButton.setLabel(label)	Button.label	N/ui/serverWidget Module	Note that label is a property.
nlobjButton.setVisible(visible)	Button.isHidden	N/ui/serverWidget Module	Note that isHidden is a property.

nlobjColumn

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjColumn	serverWidget.ListColumn	N/ui/serverWidget Module	-
nlobjColumn.addParamToURL(param, value, dynamic)	ListColumn.addParamToURL(options)	N/ui/serverWidget Module	-
nlobjColumn.setLabel(label)	ListColumn.label	N/ui/serverWidget Module	Note that label is a property.
nlobjColumn.setURL(url, dynamic)	ListColumn.setURL(options)	N/ui/serverWidget Module	-

nlobjConfiguration

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjConfiguration	record.Record	N/record Module	<p>Use the N/config Module method, <code>config.load(options)</code>, to return a <code>record.Record</code> object. Then use the <code>record.Record</code> object members to access the specified configuration page.</p> <p>For a script sample, see N/config Module Script Sample.</p>
nlobjConfiguration.getAllFields()	Record.getFields()	N/record Module	-
nlobjConfiguration.getField(<code>fldname</code>)	Record.getField(<code>options</code>)	N/record Module	-
nlobjConfiguration.getFieldText(<code>name</code>)	Record.getText(<code>options</code>)	N/record Module	-
nlobjConfiguration.getFieldTexts(<code>name</code>)	Record.getText(<code>options</code>)	N/record Module	-
nlobjConfiguration.getFieldValue(<code>name</code>)	Record.getValue(<code>options</code>)	N/record Module	-
nlobjConfiguration.getFieldValues(<code>name</code>)	Record.getValue(<code>options</code>)	N/record Module	-
nlobjConfiguration.getType()	Record.type	N/record Module	Note that type is a property.
nlobjConfiguration.setText(<code>name</code> , <code>text</code>)	Record.setText(<code>options</code>)	N/record Module	-
nlobjConfiguration.setTexts(<code>name</code> , <code>text</code>)	Record.setText(<code>options</code>)	N/record Module	-
nlobjConfiguration.setValue(<code>name</code> , <code>value</code>)	Record.setValue(<code>options</code>)	N/record Module	-
nlobjConfiguration.setValues(<code>name</code> , <code>value</code>)	Record.setValue(<code>options</code>)	N/record Module	-

nlobjContext

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjContext	<code>runtime.Script</code> <code>runtime.Session</code> <code>runtime.User</code>	N/runtime Module	-
nlobjContext.getCompany()	runtime.accountId	N/runtime Module	Note that accountId is a property.
nlobjContext.getDepartment()	User.department	N/runtime Module	Note that department is a property.
nlobjContext.getDeploymentId()	Script.deploymentId	N/runtime Module	Note that deploymentId is a property.
nlobjContext.getEmail()	User.email	N/runtime Module	Note that email is a property.
nlobjContext.getEnvironment()	runtime.envType	N/runtime Module	Note that envType is a property.
nlobjContext.getExecutionContext()	runtime.executionContext	N/runtime Module	Note that executionContext is a property.
nlobjContext.getFeature(<code>name</code>)	runtime.isFeatureInEffect(<code>options</code>)	N/runtime Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjContext.getLocation()	User.location	N/runtime Module	Note that location is a property.
nlobjContext.getLogLevel()	Script.logLevel	N/runtime Module	Note that logLevel is a property.
nlobjContext.getName()	User.name	N/runtime Module	Note that name is a property.
nlobjContext.getPercentComplete() ()	Script.percentComplete	N/runtime Module	Note that percentComplete is a property. For a script sample, see N/runtime Module Script Samples .
nlobjContext.getPermission(name)	User.getPermission(options)	N/runtime Module	-
nlobjContext.getPreference(name)	User.getPreference(options)	N/runtime Module	-
nlobjContext.getQueueCount()	runtime.queueCount	N/runtime Module	Note that queueCount is a property.
nlobjContext.getRemainingUsage() ()	Script.getRemainingUsage()	N/runtime Module	-
nlobjContext.getRole()	User.role	N/runtime Module	Note that role is a property.
nlobjContext.getRoleCenter()	User.roleCenter	N/runtime Module	Note that roleCenter is a property.
nlobjContext.getRoleId()	User.roleId	N/runtime Module	Note that roleId is a property.
nlobjContext.getScriptId()	Script.id	N/runtime Module	Note that id is a property.
nlobjContext.getSessionObject(name)	Session.get(options)	N/runtime Module	-
nlobjContext.getSetting(type, name)	Script.getParameter(options) Session.get(options) runtime.isFeatureInEffect(options) User.getPermission(options)	N/runtime Module	The method Script.getParameter(options) is equivalent to nlobjContext.getSetting('SCRIPT', name). The method Session.get(options) is equivalent to nlobjContext.getSetting('SESSION', name). The method runtime.isFeatureInEffect(options) is equivalent to nlobjContext.getSetting('FEATURE', name). The method User.getPermission(options) is equivalent to nlobjContext.getSetting('PERMISSION', name).
nlobjContext.getSubsidiary()	User.subsidiary	N/runtime Module	Note that subsidiary is a property.
nlobjContext.getUser()	User.id	N/runtime Module	Note that id is a property.
nlobjContext.getVersion()	runtime.version	N/runtime Module	Note that version is a property.
nlobjContext.setPercentComplete(pct)	Script.percentComplete	N/runtime Module	Note that percentComplete is a property.
nlobjContext.setSessionObject(name, value)	Session.set(options)	N/runtime Module	-
nlobjContext.setSetting(type, name, value)	Session.set(options)	N/runtime Module	-

nlobjCredentialBuilder

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjCredentialBuilder(string, domainString)	https.SecureString	N/https Module	-
nlobjCredentialBuilder.append(nlobjCredentialBuilder)	SecureString.appendSecureString(options) SecureString.appendString(options)	N/https Module	-
nlobjCredentialBuilder.base64()	SecureString.convertEncoding(options)	N/https Module	-
nlobjCredentialBuilder.md5()	SecureString.hash(options)	N/https Module	-
nlobjCredentialBuilder.replace(string1, string2)	-	N/https Module	There is not an equivalent of this API in SuiteScript 2.0.
nlobjCredentialBuilder.sha1()	SecureString.hash(options)	N/https Module	-
nlobjCredentialBuilder.sha256()	SecureString.hash(options)	N/https Module	-
nlobjCredentialBuilder.utf8()	SecureString.convertEncoding(options)	N/https Module	-

nlobjCSVImport

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjCSVImport	task.CsvImportTask	N/task Module	Returned by task.create(options). <pre> 1 var csvImpTaskObj = task.create({ 2 taskType: task.Task 3 Type.CSV_IMPORT, 4 //Other Params 5 }); </pre>
nlobjCSVImport.setLinkedFile(sublist, file)	CsvImportTask.linkedFiles	N/task Module	Note that linkedFiles is a property.
nlobjCSVImport.setMapping(savedImport)	CsvImportTask.mappingId	N/task Module	Note that mappingId is a property.
nlobjCSVImport.setOption(option, value)	CsvImportTask.name	N/task Module	Note that name is a property.
nlobjCSVImport.setPrimaryFile(file)	CsvImportTask.importFile	N/task Module	Note that importFile is a property.
nlobjCSVImport.setQueue(string)	CsvImportTask.queueId	N/task Module	Note that queueId is a property.

nlobjDuplicateJobRequest

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjDuplicateJobRequest	task.EntityDeduplicationTask	N/task Module	Returned by task.create(options).

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
			<pre> 1 var dedupTaskObj = task.create({ 2 taskType: task.TaskType.ENTITY_D 3 DUPLICATION, 4 //Other Params 5 }); </pre>
nlobjDuplicateJobRequest.setEntityType(entityType)	EntityDeduplicationTask.entityType	N/task Module	Note that entityType is a property.
nlobjDuplicateJobRequest.setMasterId(masterID)	EntityDeduplicationTask.masterRecordId	N/task Module	-
nlobjDuplicateJobRequest.setMasterSelectionMode(mode)	EntityDeduplicationTask.masterSelectionMode	N/task Module	Note that masterSelectionMode is a property.
nlobjDuplicateJobRequest.setOperation(operation)	EntityDeduplicationTask.dedupeMode	N/task Module	Note that dedupeMode is a property.
nlobjDuplicateJobRequest.setRecords(dupeRecords)	EntityDeduplicationTask.recordIds	N/task Module	Note that recordIds is a property.

nlobjEmailMerger

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjEmailMerger	render.EmailMergeResult	N/render Module	-
nlobjEmailMerger.merge()	See Notes	N/render Module	In SuiteScript 2.0, this is automatically called in render.mergeEmail(options) .
nlobjEmailMerger.setCustomRecord(recordType, recordId)	See Notes	N/render Module	In SuiteScript 2.0, this value is set with a render.mergeEmail(options) parameter.
nlobjEmailMerger.setEntity(entityType, entityId)	See Notes	N/render Module	In SuiteScript 2.0, this value is set with a render.mergeEmail(options) parameter.
nlobjEmailMerger.setRecipient(recipientType, recipientId)	See Notes	N/render Module	In SuiteScript 2.0, this value is set with a render.mergeEmail(options) parameter.
nlobjEmailMerger.setSupportCase(caseId)	See Notes	N/render Module	In SuiteScript 2.0, this value is set with a render.mergeEmail(options) parameter.
nlobjEmailMerger.setTransaction(transactionId)	See Notes	N/render Module	In SuiteScript 2.0, this value is set with a render.mergeEmail(options) parameter.

nlobjError

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjError	error.SuiteScriptError error.UserEventError	N/error Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjError.getCode()	SuiteScriptError.name or UserEventError.name	N/error Module	Note that SuiteScriptError.name and UserEventError.name are properties.
nlobjError.getDetails()	SuiteScriptError.message or UserEventError.message	N/error Module	Note that SuiteScriptError.message and UserEventError.message are properties.
nlobjError.getId()	SuiteScriptError.id or UserEventError.id	N/error Module	Note that SuiteScriptError.id and UserEventError.id are properties.
nlobjError.getInternalId()	UserEventError.recordId	N/error Module	Note that UserEventError.recordId is a property.
nlobjError.getStackTrace()	SuiteScriptError.stack or UserEventError.stack	N/error Module	Note that SuiteScriptError.stack and UserEventError.stack are properties.
nlobjError.getUserEvent()	UserEventError.eventType	N/error Module	Note that UserEventError.eventType is a property.

nlobjField

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjField	serverWidget.Field record.Field	N/ui/serverWidget Module N/record Module	Use the N/ui/serverWidget module to create and modify form fields in a Suitelet. Use the N/record module to access field metadata in client and server-side scripts.

nlobjFieldGroup

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjFieldGroup	serverWidget.FieldGroup	N/ui/serverWidget Module	-
nlobjFieldGroup.setCollapsible(collapsible, hidden)	FieldGroup.isCollapsible	N/ui/serverWidget Module	Note that isCollapsible is a property.
nlobjFieldGroup.setLabel(label)	FieldGroup.label	N/ui/serverWidget Module	Note that label is a property.
nlobjFieldGroup setShowBorder(show)	FieldGroup.isBorderHidden	N/ui/serverWidget Module	Note that isBorderHidden is a property.
nlobjFieldGroup.setSingleColumn(column)	FieldGroup.isSingleColumn	N/ui/serverWidget Module	Note that isSingleColumn is a property.

nlobjFile

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjFile	file.File	N/file Module	For a script sample, see N/file Module Script Sample.
nlobjFile.getDescription()	File.description	N/file Module	Note that description is a property.
nlobjFile.getFolder()	File.folder	N/file Module	Note that folder is a property. For a script sample, see N/file Module Script Sample.
nlobjFile.getId()	File.id	N/file Module	Note that id is a property. For a script sample, see N/file Module Script Sample.
nlobjFile.getName()	File.name	N/file Module	Note that name is a property. For a script sample, see N/file Module Script Sample.
nlobjFile.getSize()	File.size	N/file Module	Note that size is a property.
nlobjFile.getType()	File.fileType	N/file Module	Note that fileType is a property. For a script sample, see N/file Module Script Sample. For a script sample, see N/file Module Script Sample.
nlobjFile.getURL()	File.url	N/file Module	Note that url is a property.
nlobjFile.getValue()	File.getContents()	N/file Module	-
nlobjFile.isInactive()	File.isInactive	N/file Module	Note that isInactive is a property.
nlobjFile.isOnline()	File.isOnline	N/file Module	Note that isOnline is a property. For a script sample, see N/file Module Script Sample.
nlobjFile.setDescription(description)	File.description	N/file Module	Note that description is a property.
nlobjFile.setEncoding(encodingType)	File.encoding	N/file Module	Note that encoding is a property.
nlobjFile.setFolder(id)	File.folder	N/file Module	Note that folder is a property. You can also set the folder during file creation with <code>file.create(options)</code> . For a script sample, see N/file Module Script Sample.
nlobjFile.setIsInactive(inactive)	File.isInactive	N/file Module	Note that isInactive is a property.
nlobjFile.setIsOnline(online)	File.isOnline	N/file Module	Note that isOnline is a property.

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
			For a script sample, see N/file Module Script Sample .
nlobjFile.setName(name)	File.name	N/file Module	Note that name is a property. For a script sample, see N/file Module Script Sample .

nlobjForm

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjForm	serverWidget.Form	N/ui/serverWidget Module	-
nlobjForm.addButton(name, label, script)	Form.addButton(options)	N/ui/serverWidget Module	-
nlobjForm.addCredentialField(id, label, website, scriptId, value, entityMatch, tab)	Form.addCredentialField(options)	N/ui/serverWidget Module	-
nlobjForm.addField(name, type, label, sourceOrRadio, tab)	Form.addField(options)	N/ui/serverWidget Module	For a script sample, see N/ui/serverWidget Module Script Samples
nlobjForm.addFieldGroup(name, label, tab)	Form.addFieldGroup(options)	N/ui/serverWidget Module	-
nlobjForm.addPageLink(type, title, url)	Form.addPageLink(options)	N/ui/serverWidget Module	-
nlobjForm.addResetButton(label)	Form.addResetButton(options)	N/ui/serverWidget Module	-
nlobjForm.addSubList(name, type, label, tab)	Form.addSublist(options)	N/ui/serverWidget Module	For a script sample, see N/ui/serverWidget Module Script Samples
nlobjForm.addSubmitButton(label)	Form.addSubmitButton(options)	N/ui/serverWidget Module	For a script sample, see N/ui/serverWidget Module Script Samples
nlobjForm.addSubTab(name, label, tab)	Form.addSubtab(options)	N/ui/serverWidget Module	-
nlobjForm.addTab(name, label)	Form.addTab(options)	N/ui/serverWidget Module	-
nlobjForm.getButton(name)	Form.getButton(options)	N/ui/serverWidget Module	-
nlobjForm.getField(name, radio)	Form.getField(options)	N/ui/serverWidget Module	-
nlobjForm.getSubList(name)	Form.getSublist(options)	N/ui/serverWidget Module	-
nlobjForm.getSubTab(name)	Form.getSubtab(options)	N/ui/serverWidget Module	-
nlobjForm.getTab(name)	Form.getTab(options)	N/ui/serverWidget Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjForm.getTabs()	Form.getTabs()	N/ui/serverWidget Module	-
nlobjForm.insertField(field, nextfld)	Form.insertField(options)	N/ui/serverWidget Module	-
nlobjForm.insertSubList(sublist, nextsub)	Form.insertSublist(options)	N/ui/serverWidget Module	-
nlobjForm.insertSubTab(subtab, nextsub)	Form.insertSubtab(options)	N/ui/serverWidget Module	-
nlobjForm.insertTab(tab, nexttab)	Form.insertTab(options)	N/ui/serverWidget Module	-
nlobjForm.removeButton(name)	Form.removeButton(options)	N/ui/serverWidget Module	-
nlobjForm.setFieldValues(values)	Form.updateDefaultValues(options)	N/ui/serverWidget Module	-
nlobjForm.setScript(script)	Form.clientScriptFileDialog Form.clientScriptModulePath	N/ui/serverWidget Module	Note that clientScriptFileDialog and clientScriptModulePath are properties. Use one of these SuiteScript 2.0 properties to attach an ad hoc client script to a form.
nlobjForm.setTitle(title)	Form.title	N/ui/serverWidget Module	Note that title is a property.

nlobjFuture

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjFuture	task.EntityDeduplicationTaskStatus	N/task Module	-
nlobjFuture.isCancelled()	EntityDeduplicationTaskStatus.status	N/task Module	Note that status is a property.
nlobjFuture.isDone()	EntityDeduplicationTaskStatus.status	N/task Module	Note that status is a property.

nlobjJobManager

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjJobManager	task.EntityDeduplicationTaskStatus	N/task Module	-
nlobjJobManager.createJobRequest()	task.create(options)	N/task Module	-
nlobjJobManager.getFuture()	task.checkStatus(options)	N/task Module	-
nlobjJobManager.submit(nlobjDuplicateJobRequest)	EntityDeduplicationTask.submit()	N/task Module	-

nlobjList

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjList	serverWidget.List	N/ui/serverWidget Module	-
nlobjList.addButton(name, label, script)	List.addButton(options)	N/ui/serverWidget Module	-
nlobjList.addColumn(name, type, label, align)	List.addColumn(options)	N/ui/serverWidget Module	-
nlobjList.addEditColumn(column, showView, showHrefCol)	List.addEditColumn(options)	N/ui/serverWidget Module	-
nlobjList.addPageLink(type, title, url)	List.addPageLink(options)	N/ui/serverWidget Module	-
nlobjList.addRow(row)	List.addRow(options)	N/ui/serverWidget Module	-
nlobjList.addRows(rows)	List.addRows(options)	N/ui/serverWidget Module	-
nlobjList.setScript(script)	List.clientScriptFileDialog List.clientScriptModulePath	N/ui/serverWidget Module	Note that clientScriptFileDialog and clientScriptModulePath are properties. Use one of these SuiteScript 2.0 properties to attach an ad hoc client script to a form.
nlobjList.setStyle(style)	List.style	N/ui/serverWidget Module	Note that style is a property.
nlobjList.setTitle(title)	List.title	N/ui/serverWidget Module	Note that title is a property.

nlobjLogin

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjLogin	See Notes.	N/auth Module	See nlobjLogin members.
nlobjLogin.changeEmail (currentPassword, newEmail, justThisAccount)	auth.changeEmail(options)	N/auth Module	For a script sample, see N/auth Module Script Sample .
nlobjLogin.changePassword (currentPassword, newPassword)	auth.changePassword(options)	N/auth Module	For a script sample, see N/auth Module Script Sample

nlobjMergeResult

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjMergeResult	render.EmailMergeResult	N/render Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjMergeResult.getBody()	EmailMergeResult.body	N/render Module	Note that body is a property.
nlobjMergeResult.getSubject()	EmailMergeResult.subject	N/render Module	Note that subject is a property.

nlobjPortlet

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjPortlet	Portlet Object	See Notes	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.addColumn(name, type, label, just)	Portlet.addColumn(options)	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.addEditColumn(column, showView, showHrefCol)	Portlet.addEditColumn(options)	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.addField(name, type, label, source)	Portlet.addField(options)	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.addLine(text, url, indent)	Portlet.addLine(options)	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.addRow(row)	Portlet.addRow(options)	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.addRows(rows)	Portlet.addRows(options)	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.setHtml(html)	Portlet.html	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.setRefreshInterval(n)	-	-	There is no SuiteScript 2.0 direct equivalent for this method. For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjPortlet.setScript(scriptid)	Portlet.clientScriptFileId Portlet.clientScriptModulePath	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.setSubmitButton(url, label, target)	Portlet.setSubmitButton(options)	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .
nlobjPortlet.setTitle(title)	Portlet.title	-	For additional information, see the help topic SuiteScript 2.0 Portlet Script Type .

nlobjRecord

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjRecord	record.Record currentRecord.CurrentRecord	N/record Module N/currentRecord Module	-
nlobjRecord.commitLineItem(group, ignoreRecalc)	Record.commitLine(options) CurrentRecord.commitLine(options)	N/record Module N/currentRecord Module	-
nlobjRecord.createCurrentLineItemSubrecord(sublist, fldname)	Record.getCurrentSublistSubrecord(options) CurrentRecord.getCurrentSublistSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlobjRecord.createSubrecord(fldname)	Record.getSubrecord(options) CurrentRecord.getSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlobjRecord.editCurrentLineItemSubrecord(sublist, fldname)	Record.getCurrentSublistSubrecord(options) CurrentRecord.getCurrentSublistSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlobjRecord.editSubrecord(fldname)	Record.getSubrecord(options) CurrentRecord.getSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlobjRecord.findLineItemMatrixValue(group, fldnam, column, val)	Record.findMatrixSublistLineWithValue(options)	N/record Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
	CurrentRecord.findMatrixSublistLineWithValue(options)	N/currentRecord Module	
nlobjRecord.findLineItemValue(group, fldnam, value)	Record.findSublistLineWithValue(options) CurrentRecord.findSublistLineWithValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getAllFields()	Record.getFields()	N/record Module	-
nlobjRecord.getAllLineItemFields(group)	Record.getSublistFields(options)	N/record Module	-
nlobjRecord.getCurrentLineItemDateTimeValue(type, fieldId, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlobjRecord.getCurrentLineItemMatrixValue(group, fldnam, column)	Record.getCurrentMatrixSublistValue(options) CurrentRecord.getCurrentMatrixSublistValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getCurrentLineItemValue(type, fldnam)	Record.getCurrentSublistValue(options) CurrentRecord.getCurrentSublistValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getCurrentLineItemValues(type, fldnam)	Record.getCurrentSublistValue(options) CurrentRecord.getCurrentSublistValue(options)	N/record Module N/currentRecord Module	Method returns an array for multi-select fields.
nlobjRecord.getDateTimeValue(fieldId, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlobjRecord.getField(fldnam)	Record.getField(options) CurrentRecord.getField(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getText(name)	Record.getText(options) CurrentRecord.getText(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getTexts(name)	Record.getText(options) CurrentRecord.getText(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getValue(name)	Record.getValue(options) CurrentRecord.getValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getValues(name)	Record.getValue(options) CurrentRecord.getValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getId()	Record.id	N/record Module	-
nlobjRecord.getLineItemCount(group)	Record.getLineCount(options)	N/record Module	-
nlobjRecord.getLineItemDateValue(type, fieldId, lineNum, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjRecord.getLineItemField(group, fldnam, linenum)	Record.getSublistField(options) CurrentRecord.getSublistField(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getLineItemMatrixField(group, fldnam, linenum, column)	Record.getMatrixSublistField(options) CurrentRecord.getMatrixSublistField(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getLineItemMatrixValue(group, fldnam, lineum, column)	Record.getMatrixSublistValue(options) CurrentRecord.getMatrixSublistValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getLineItemText(group, fldnam, linenum)	Record.getSublistText(options) CurrentRecord.getSublistText(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getLineItemValue(group, name, linenum)	Record.getSublistValue(options) CurrentRecord.getSublistValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getLineItemValues(type, fldnam, linenum)	Record.getSublistValue(options) CurrentRecord.getSublistValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getMatrixCount(group, fldnam)	Record.getMatrixHeaderCount(options) CurrentRecord.getMatrixHeaderCount(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getMatrixField(group, fldname, column)	Record.getMatrixHeaderField(options) CurrentRecord.getMatrixHeaderField(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getMatrixValue(group, fldnam, column)	Record.getMatrixHeaderValue(options) CurrentRecord.getMatrixHeaderValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.getRecordType()	Record.type	N/record Module	-
nlobjRecord.insertLineItem(group, linenum, ignoreRecalc)	Record.insertLine(options) CurrentRecord.insertLine(options)	N/record Module N/currentRecord Module	-
nlobjRecord.removeLineItem(group, linenum, ignoreRecalc)	Record.removeLine(options) CurrentRecord.removeLine(options)	N/record Module N/currentRecord Module	-
nlobjRecord.removeCurrentLineItemSubrecord(sublist, fldname)	Record.removeCurrentSublistSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlobjRecord.removeSubrecord(fldname)	Record.removeSubrecord(options)	N/record Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
	CurrentRecord.removeSubrecord(options)	N/currentRecord Module	in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlobjRecord.selectLineItem(group, linenum)	Record.selectLine(options) CurrentRecord.selectLine(options)	N/record Module N/currentRecord Module	-
nlobjRecord.selectNewLineItem(group)	Record.selectNewLine(options) CurrentRecord.selectNewLine(options)	N/record Module N/currentRecord Module	-
nlobjRecord.setCurrentLineItemDateTimeValue(type, fieldId, dateTime, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlobjRecord.setCurrentLineItemMatrixValue(group, fldnam, column, value)	Record.setCurrentMatrixSublistValue(options) CurrentRecord.setCurrentMatrixSublistValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.setCurrentLineItemValue(group, name, value)	Record.setCurrentSublistValue(options) CurrentRecord.setCurrentSublistValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.setDate TValue(fieldId, dateT, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlobjRecord.setFieldText(name, text)	Record.setText(options) CurrentRecord.setText(options)	N/record Module N/currentRecord Module	-
nlobjRecord.setFieldTexts(name, text)	Record.setText(options) CurrentRecord.setText(options)	N/record Module N/currentRecord Module	-
nlobjRecord.setFieldValue(name, value)	Record.setValue(options) CurrentRecord.setValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.setFieldValues(name, value)	Record.setValue(options) CurrentRecord.setValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.setLineItemDate TValue(type, fieldId, lineNum, dateT, timeZone)	See Notes	N/format Module	Use the N/format module to mimic this functionality in SuiteScript 2.0.
nlobjRecord.setLineItemValue(group, name, linenum, value)	Record.setSublistValue(options)	N/record Module	-
nlobjRecord.setMatrixValue(group, fldnam, column, value)	Record.setMatrixHeaderValue(options) CurrentRecord.setMatrixHeaderValue(options)	N/record Module N/currentRecord Module	-
nlobjRecord.viewCurrentLineItemSubrecord(sublist, fldname)	Record.getCurrentSublistSubrecord(options) CurrentRecord.getCurrentSublistSubrecord(options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
			topics under SuiteScript 2.0 Scripting Subrecords .
nlobjRecord.viewLineItemSubrecord (sublist, fldname, linenum)	Record.getSublistSubrecord (options)	N/record Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .
nlobjRecord.viewSubrecord (fldname)	Record.getSubrecord(options) CurrentRecord.getSubrecord (options)	N/record Module N/currentRecord Module	Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .

nlobjRequest

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjRequest	http.ServerRequest	N/http Module N/ https Module	-
nlobjRequest.getAllHeaders()	ServerRequest.headers	N/http Module N/ https Module	ServerRequest.headers(options) is read-only.
nlobjRequest.getAllParameters()	ServerRequest.parameters	N/http Module N/ https Module	ServerRequest.parameters(options) is read-only.
nlobjRequest.getBody()	ServerRequest.body	N/http Module N/ https Module	ServerRequest.body(options) is read-only.
nlobjRequest.getFile(id)	ServerRequest.files	N/http Module N/ https Module	ServerRequest.files(options) is read-only.
nlobjRequest.getHeader(name)	ServerRequest.headers	N/http Module N/ https Module	ServerRequest.headers(options) is read-only.
nlobjRequest.getLineItemCount (group)	ServerRequest.getLineCount (options)	N/http Module N/ https Module	-
nlobjRequest.getLineItemValue (group, name, line)	ServerRequest.getSublistValue (options)	N/http Module N/ https Module	-
nlobjRequest.getMethod()	ServerRequest.method	N/http Module N/ https Module	ServerRequest.method(options) is read-only.
nlobjRequest.getParameter(name)	ServerRequest.parameters	N/http Module N/ https Module	ServerRequest.parameters(options) is read-only.
nlobjRequest.getParameterValues (name)	ServerRequest.parameters	N/http Module N/ https Module	ServerRequest.parameters(options) is read-only.
nlobjRequest.getURL()	ServerRequest.url	N/http Module N/ https Module	ServerRequest.url is read-only.

nlobjResponse

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjResponse	http.ServerResponse	N/http Module N/ https Module	-
nlobjResponse.addHeader(name, value)	ServerResponse.addHeader (options)	N/http Module N/ https Module	-
nlobjResponse.getAllHeaders()	ServerResponse. getHeader(options) or ServerResponse.headers	N/http Module N/ https Module	-
nlobjResponse.getBody()	ClientResponse.body	N/http Module N/ https Module	Note that ClientResponse .body is a property.
nlobjResponse.getCode()	ClientResponse.code	N/http Module N/ https Module	Note that ClientResponse .code is a property.
nlobjResponse.getError()	See Notes	N/http Module N/ https Module	There is no SuiteScript 2.0 equivalent for this method.
nlobjResponse.getHeader(name)	ServerResponse. getHeader(options)	N/http Module N/ https Module	-
nlobjResponse.getHeaders(name)	ServerResponse. getHeader(options)	N/http Module N/ https Module	If multiple values are assigned to the header name,serverResponse. getHeader(options) returns the values as an Array.
nlobjResponse.renderPDF (xmlString)	ServerResponse. renderPdf(options)	N/http Module N/ https Module	-
nlobjResponse.setCDNCacheable (type)	ServerResponse. setCdnCacheable(options)	N/http Module N/ https Module	-
nlobjResponse.setContentType(type name, disposition)	See Notes	N/http Module N/ https Module	There is no direct equivalent for this method in SuiteScript 2.0.
nlobjResponse.setEncoding (encodingType)	See Notes	N/http Module N/ https Module	There is no direct equivalent for this method in SuiteScript 2.0.
nlobjResponse.setHeader(name, value)	ServerResponse. setHeader(options)	N/http Module N/ https Module	-
nlobjResponse.sendRedirect(type, identifier, id, editmode, parameters)	ServerResponse.sendRedirect (options)	N/http Module N/ https Module	-
nlobjResponse.write(output)	ServerResponse.write(options)	N/http Module N/ https Module	-
nlobjResponse.writeLine(output)	ServerResponse. writeLine(options)	N/http Module N/ https Module	-
nlobjResponse.writePage (pageobject)	ServerResponse. writePage(options)	N/http Module N/ https Module	-

nlobjSearch

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSearch	search.Search	N/search Module	-
nlobjSearch.addColumn(column)	Search.columns	N/search Module	Note that Search.columns is a property.
nlobjSearch.setColumns(columns)	Search.columns	N/search Module	Note that Search.columns is a property.
nlobjSearch.addFilter(filter)	Search.filters	N/search Module	Note that Search.filters is a property.
nlobjSearch.addFilters(filters)	Search.filters	N/search Module	Note that Search.filters is a property.
nlobjSearch.deleteSearch()	search.delete(options)	N/search Module	For a script sample, see N/search Module Script Samples .
nlobjSearch.getColumns()	Search.columns	N/search Module	Note that columns is a property.
nlobjSearch.getFilterExpression()	Search.filterExpression	N/search Module	Note that filterExpression is a property.
nlobjSearch.getFilters()	Search.filters	N/search Module	Note that filters is a property.
nlobjSearch.getId()	Search.searchId	N/search Module	Note that id is a property.
nlobjSearch.getIsPublic()	Search.isPublic	N/search Module	Note that isPublic is a property.
nlobjSearch.getscriptId()	Search.id	N/search Module	-
nlobjSearch.getSearchType()	Search.searchType	N/search Module	Note that searchType is a property.
nlobjSearch.runSearch()	Search.run()	N/search Module	-
nlobjSearch.saveSearch(title, scriptId)	Search.save()	N/search Module	-
nlobjSearch.setColumns(columns)	Search.columns	N/search Module	Note that columns is a property.
nlobjSearch.setFilterExpression(filterExpression)	Search.filterExpression	N/search Module	Note that filterExpression is a property.
nlobjSearch.setFilters(filters)	Search.filters	N/search Module	Note that filters is a property.
nlobjSearch.setIsPublic(type)	Search.isPublic	N/search Module	Note that isPublic is a property.
nlobjSearch.setRedirectURLToSearch()	redirect.toSavedSearch(options) redirect.toSearch(options)	N/redirect Module	-
nlobjSearch.setRedirectURLToSearchResults()	redirect.toSearchResult(options) redirect.toSavedSearchResult(options)	N/redirect Module	-

nlobjSearchColumn

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSearchColumn	search.Column	N/search Module	-

nlobjSearchFilter

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSearchFilter	search.Filter	N/search Module	-

nlobjSearchResult

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSearchResult	search.Result	N/search Module	-

nlobjSearchResultSet

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSearchResultSet	search.ResultSet	N/search Module	-

nlobjSelectOption

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSelectOption	See Notes	N/record Module	See mapping for nlobjSelectOption methods.
nlobjSelectOption.getId()	N/record: Field. getSelectOptions(options) N/currentRecord: Field. getSelectOptions(options)	N/record Module N/currentRecord Module	-
nlobjSelectOption.getText()	N/record: Field. getSelectOptions(options) N/currentRecord: Field. getSelectOptions(options)	N/record Module N/currentRecord Module	-

nlobjSublist

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSubList	serverWidget.Sublist	N/ui/serverWidget Module	-
nlobjSublist.addButton(name, label, script)	Sublist.addButton(options)	N/ui/serverWidget Module	-
nlobjSublist.addField(name, type, label, source)	Sublist.addField(options)	N/ui/serverWidget Module	-
nlobjSublist.addMarkAllButtons()	Sublist.addMarkAllButtons()	N/ui/serverWidget Module	-

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSublist.addRefreshButton()	Sublist.addRefreshButton()	N/ui/serverWidget Module	-
nlobjSublist.getLineItemCount()	Sublist.lineCount	N/ui/serverWidget Module	Note that lineCount is a property
nlobjSublist.getLineItemValue (group, fldnam, linenum)	Sublist.getSublistValue(options)	N/ui/serverWidget Module	-
nlobjSublist.setAmountField(field)	Sublist.updateTotallingFieldId (options)	N/ui/serverWidget Module	-
nlobjSublist.setDisplayType(type)	Sublist.displayType	N/ui/serverWidget Module	Note that displayType is a property
nlobjSublist.setHelpText(help)	Sublist.helpText	N/ui/serverWidget Module	Note that helpText is a property
nlobjSublist.setLabel(label)	Sublist.label	N/ui/serverWidget Module	Note that label is a property
nlobjSublist.setLineItemValue (name, linenum,value)	Sublist.setSublistValue(options)	N/ui/serverWidget Module	-
nlobjSublist.setLineItemValues (values)	See Notes	N/ui/serverWidget Module	There is not a SuiteScript 2.0 direct equivalent for this method.
nlobjSublist.setUniqueField(name)	Sublist.updateUniqueFieldId (options)	N/ui/serverWidget Module	Note that uniqueFieldId is a property

nlobjSubrecord

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjSubrecord	See Notes	N/record Module	<p>SuiteScript 2.0 subrecords are returned as record.Record objects.</p> <p>Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords.</p>
nlobjSubrecord.cancel()	See Notes	N/record Module	<p>SuiteScript 2.0 subrecords are returned as record.Record objects.</p> <p>Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords.</p>
nlobjSubrecord.commit()	See Notes	N/record Module	<p>This API does not have a SuiteScript 2.0 equivalent. SuiteScript 2.0 subrecords are returned as record.Record objects.</p> <p>Note that scripting subrecords in SuiteScript 2.0 is fundamentally different from scripting subrecords in SuiteScript 1.0. For additional</p>

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
			information, see the SuiteScript 2.0 topics under SuiteScript 2.0 Scripting Subrecords .

nlobjTab

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjTab	serverWidget.Tab	N/ui/serverWidget Module	-
nlobjTab.setLabel(label)	Tab.label	N/ui/serverWidget Module	Note that label is a property
nlobjTab.setHelpText(help)	Tab.helpText	N/ui/serverWidget Module	Note that helpText is a property

nlobjTemplateRenderer

SuiteScript 1.0 API	SuiteScript 2.0 API	SuiteScript 2.0 Module	Notes
nlobjTemplateRenderer	render.TemplateRenderer	N/render Module	For a script sample, see N/render Module Script Sample .
nlobjTemplateRenderer.addRecord(var, record)	TemplateRenderer.addRecord(options)	N/render Module	For a script sample, see N/render Module Script Sample .
nlobjTemplateRenderer.addSearchResults(searchResult)	TemplateRenderer.addSearchResults(options)	N/render Module	For a script sample, see N/render Module Script Sample .
nlobjTemplateRenderer.renderToResponse()	TemplateRenderer.renderToResponse(options)	N/render Module	-
nlobjTemplateRenderer.renderToString()	TemplateRenderer.renderAsString()	N/render Module	-
nlobjTemplateRenderer.setTemplate(template)	TemplateRenderer.templateContent	N/render Module	For a script sample, see N/render Module Script Sample .

SuiteScript 2.0 Global Objects



Note: The content in this help topic pertains to SuiteScript 2.0.

SuiteScript 2.0 includes the following global objects. You can use these objects in your scripts without loading them as dependencies.

- [define Object](#)
- [require Function](#)
- [log Object](#)
- [util Object](#)
- [toString\(\)](#)
- [JSON object](#)
- [Promise Object](#)



Note: In JavaScript, all functions are objects. The [define Object](#) and [require Function](#) topics discuss the `define()` and `require()` functions used by SuiteScript 2.0 to load and define modules.

define Object



Note: The content in this help topic pertains to SuiteScript 2.0.

The `define` object is an overloaded function that is used to create entry point scripts and custom modules in SuiteScript 2.0. This function executes asynchronously on the client side and synchronously on the server side. The `define` object conforms to the [Asynchronous Module Definition \(AMD\)](#) specification.



Note: An overloaded function has multiple signatures. A signature is the function name and all available parameters.

SuiteScript 2.0 supports the following `define()` signatures:

Type	Name	Return Type / Value Type	Description
Function	define(moduleObject)	object	Returns a module object based on the supplied <code>moduleObject</code> argument. The <code>moduleObject</code> argument can be any JavaScript object, including a function. Use this <code>define()</code> signature if your entry point script or custom module requires no dependencies.
	define(id, [dependencies,] moduleObject)	object	Loads all dependencies and then executes the supplied callback function. Returns a module object based on the callback.

Use the `define()` function to do the following:

- Create a SuiteScript script file. Load the required dependent modules and define the functionality for the SuiteScript script type in the callback function. The return statement in the callback function must include at least one entry point and entry point function. All entry points must belong to the same script type.

Any implementation of a SuiteScript script type that returns an entry point must use the `define()` function.

- Create and return a custom module. The custom module can then be used as dependency in another script. Use the `define(id, [dependencies,] moduleObject)` signature if your module requires dependencies. If the custom module does not require any dependencies, use the `define(moduleObject)` signature.

For more information about custom modules, see the help topic [SuiteScript 2.0 Custom Modules](#).

For more information about entry points, see the help topic [SuiteScript 2.0 Script Types](#).

define() Function Guidelines

Use the following guidelines with the `define()` function:

- SuiteScript API calls can be executed only after the `define` callback's return statement has executed. Consequently, you cannot use native SuiteScript 2.0 module methods when you *create* a custom module. You can make SuiteScript API calls after the Module Loader creates and loads the custom module.
- If you need to debug your code on demand in the SuiteScript Debugger, you must use a `require()` Function. The SuiteScript Debugger cannot step though a `define()` function.
- All dependencies used in the `define()` function are loaded before the callback function executes.
- You can load only modules that are stored in the NetSuite File Cabinet. Do not attempt to import scripts via HTTP or HTTPS.

For example, if you specify `define(['http://somewebsite.com/purchaseTotal.js'], function(purchaseTotal){...});`, the `purchaseTotal` dependency is not valid.

define(moduleObject)

 Note: The content in this help topic pertains to SuiteScript 2.0.	
Description	<p>Function used to create entry point scripts and custom modules in SuiteScript 2.0. For more information, see the help topics SuiteScript 2.0 Entry Point Script Creation and Deployment and SuiteScript 2.0 Custom Modules.</p> <p>Use this <code>define()</code> signature if your entry point script or custom module requires no dependencies.</p> <p>If you are creating an entry point script, the <code>define()</code> function must return an object consisting of at least one key:value pair. Each key must be an entry point and the corresponding value must be a named entry point function. All entry points must be for the same script type. Your script can have only one entry point script and the entry point script must be only one script type.</p>
Returns	Object
Global object	<code>define Object</code>
Since	Version 2015 Release 2

Parameters

Parameter	Type	Required / Optional	Description	Since
moduleObject	Object	required	A callback function or a module object.	Version 2015 Release 2

Syntax

The following code show sample syntax for the define(moduleobject) function signature. These code samples are not functional examples and do not provide complete list of ways this define() signature can be used.

Define a Function

```

1 // lib.js
2 define({
3     test: function () {
4         return true;
5     }
6 });
7 );
```

OR

```

1 // lib.js
2 define(function () {
3 {
4     return true
5 });
6 );
```

Define an object

```

1 // lib.js
2 define({
3     color: "black",
4     size: "unisize"
5 });
```

Define a Primitive Value

```

1 // lib.js
2 define("test");
```

define(id, [dependencies,] moduleObject)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Function used to create entry point scripts and custom modules in SuiteScript 2.0. For more information, see the help topics SuiteScript 2.0 Entry Point Script Creation and Deployment and SuiteScript 2.0 Custom Modules . Use this define() signature if your entry point script or custom module has required dependencies.
---------------------------	--

	If you are creating an entry point script, the define() function must return an object consisting of at least one key:value pair. Each key must be an entry point and the corresponding value must be a named entry point function. All entry points must be for the same script type. Your script can have only one entry point script and the entry point script must be only one script type. Your entry point script can, however, load multiple custom modules as dependencies. There is no limit to the number of dependencies your entry point script can load.
Returns	Object
Global object	define Object
Since	Version 2015 Release 2

Parameters

Parameter	Type	Required / Optional	Description	Since
id	string	optional	Defines the id of the module.	Version 2015 Release 2
dependencies	string []	optional	<p>Represents all module dependencies required by the callback function.</p> <p>Use the following syntax:</p> <ul style="list-style-type: none"> ■ Native SuiteScript 2.0 modules: ['N/<module name>'] ■ Custom modules: ['/<path to module file in File Cabinet>/<module name>'] <p>For other options, see the help topic Module Dependency Paths.</p>	Version 2015 Release 2
moduleObject	Function Object	required	A callback function or a module object	Version 2015 Release 2

Errors

Error Code	Message	Thrown If
MODULE_DOES_NOT_EXIST	Module does not exist: {module path/name}	<p>The NetSuite module or custom module listed as a dependency does not exist.</p> <p>Note that NetSuite only reports the first error encountered. If you have multiple dependent modules and you receive this error, verify that all module paths and names are correct.</p>

Syntax for Module ID

The following code shows sample syntax for the define(id, [dependencies,] callback) function signature. These code samples are not functional examples and do not provide complete list of ways this define() signature can be used.

```

1 ...
2 define('mymodule', ['/test', '/sample'], function(test, sample){...});
3 ...

```

Syntax for Entry Point Script

The following example is a SuiteScript user event script type that creates a Phone Call record on the `afterSubmit` trigger.

```

1  /**
2  * @NApiVersion 2.x
3  * @NScriptType UserEventScript
4  */
5
6 define(['N/record'],
7   function (record)
8   {
9     function createPhoneCall(context)
10    {
11      if (context.type !== context.UserEventTypes.CREATE)
12        return;
13      var customerRecord = context.newRecord();
14      if (customerRecord.getValue('salesrep'))
15      {
16        var phoneCall = record.create({
17          type: record.Type.PHONE_CALL,
18          isDynamic: true
19        });
20        phoneCall.setValue({
21          fieldId: 'title',
22          value: 'Make follow-up call to new customer'
23        });
24        phoneCall.setValue('assigned', customerRecord.getValue('salesrep'));
25        phoneCall.setValue('phone', customerRecord.getValue('phone'));
26        try
27        {
28          var callId = phoneCall.save();
29          log.debug({
30            title: 'Call record created successfully',
31            details: 'Id: ' + callId
32          });
33        }
34        catch (e)
35        {
36          log.error(e.name);
37        }
38      }
39      return {
40        afterSubmit: createPhoneCall
41      };
42    });
43 });

```

Syntax for Custom Module

The following code shows the syntax for creating a custom SuiteScript 2.0 module in the script file `lib.js`.

```

1 // lib.js
2 define ['./api/bar'], function(bar){ // require bar custom module
3   return {
4     makeSomething: function(){ // define function lib.makeSomething()
5       var barObj = bar.create(); // use create() function from bar custom module
6       return bar.convertToThing(); // returns the value of bar module function convertToThing()
7     }
8   };
9 });

```

The following code shows the syntax for calling the function `lib` from the custom module `test.js` in a separate script file:

```
1 // test.js
```

```

2 | require(['lib'], function (lib) { // require custom module (defined above)
3 |
4 |     return lib.makeSomething(); // return value of makeSomething function in custom module
5 |
6 });

```

require Function

Note: The content in this help topic pertains to SuiteScript 2.0.

The require function is a global object that implements the require() Module Loader interface for SuiteScript 2.0. It conforms to the Asynchronous Module Definition (AMD) specification. When NetSuite executes the require() function, it executes the callback function and loads the dependencies when they are needed.

This function executes asynchronously on the client side and synchronously on the server side.

Note: Only use the require() function if you want to loading an existing module. If you want to create an entry point script or a new custom module, use the [define Object](#).

Use the require() function to achieve progressive loading of native SuiteScript 2.0 modules and custom modules. When you use the require() function, dependencies are not loaded until they are needed. This can help increase script performance.

For example, if you add lib1 as a dependency. When you call a method that is part of lib1, the Module Loader loads the module and executes the method. See [Syntax](#).

Type	Name	Return Type / Value Type	Description
Function	require([dependencies,] callback)	Void	Loads a SuiteScript 2.0 entry point script or a SuiteScript 2.0 custom module. Executes the callback function and loads the dependencies when they are required.

Note: To configure a require Object, you can associate a script to a JSON configuration file using a JSDoc tag. This is helpful to configure loading of a custom module. Properties that can hold feature metadata, aliases, paths, package, and mapping information related to a module id are supported. See [require Configuration](#).

require([dependencies,] callback)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Function used to load a module only when the module is needed. When NetSuite executes the require() function, it executes the callback function and loads the dependencies when they are required. If you add a module as a dependency and the module is never used, the dependency is never loaded.
---------------------------	---

	 Note: This function conforms to the Asynchronous Module Definition (AMD) specification. For more information, see require Function .
Returns	void
Global object	require Function
Since	Version 2015 Release 2

Parameters

Parameter	Type	Required / Optional	Description	Since
dependencies	string []	optional	<p>Represents all module dependencies required by the callback function.</p> <p>Use the following syntax:</p> <ul style="list-style-type: none"> ■ Native SuiteScript 2.0 modules: ['N/<module name>'] ■ Custom modules: ['/<path to module file in File Cabinet>/<module name>'] <p>See the help topic Module Dependency Paths.</p>	Version 2015 Release 2
callback	Function	required	Callback function to execute. Dependent modules are not loaded until they are required.	Version 2015 Release 2

Errors

Error Code	Message	Thrown If
MODULE_DOES_NOT_EXIST	Module does not exist: {module path/name}	<p>The NetSuite module or custom module dependency does not exist.</p> <p>Note that NetSuite only reports the first error encountered. If you have multiple dependent modules and you receive this error, verify that all module paths and names are correct.</p>

Syntax

The following example shows progressive loading of modules in a script. For a functional example, see [require Function](#).

```

1 define({
2   newInstance: function (type)
3   {
4     switch (type)
5     {
6       case 'lib1' :
7         require(['/lib1'], function (lib1) //Module Loader loads lib1
8         {
9           return new lib1();

```

```

10      })
11      break;
12  case 'lib2' :
13    require(['lib2'], function (lib2) //Module Loader loads lib2
14    {
15      return new lib2();
16    })
17    break;
18  default :
19    return null;
20  }
21 });
22 });

```

require Configuration

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

SuiteScript provides advanced options that provide you with greater control over require configuration.

If you set up a valid @NAmConfig JSDoc tag, SuiteScript implements the require configuration settings *before* loading dependencies. Configure the require Object before loading dependences so that you can run multiple client scripts with different configurations. Using the JSDoc tag can also support re-use by letting you use a common configuration across multiple scripts.

To configure a require Object, do the following:

- Add the @NAmConfig tag and provide a File Cabinet path to the configuration file

```

1 /**
2  * @NAmConfig /SuiteScripts/configuration.json
3 */

```

- SuiteScript will require a custom entry point module and its dependencies using the AMD configuration. For a list of supported configuration parameters, see [require Configuration Parameters](#). Your require configuration must be in JSON format. For example:

```

1 {
2   "baseUrl" : "/SuiteBundles"
3 }

```



Important: Ensure that configuration file uses JSON syntax (and not JavaScript syntax). For more information about JSON, visit <http://json.org/>.

You can use [JSON.stringify\(obj\)](#) to convert a JavaScript object value to a key-value pair string in JSON form.

require Configuration Parameters

SuiteScript accepts the configuration values outlined at <https://github.com/amdjs/amdjs-api/blob/master/CommonConfig.md> (version fd45c71).

You can use the JS Doc tag to point a configuration file that holds the configuration values, such as when you want to set properties before loading a custom module, or set up configuration for improved compatibility.

The following configuration parameters are supported for require Object configuration:

Parameter	Type	Required / Optional	Sample Usage	Since
baseUrl	string	optional	<ul style="list-style-type: none"> ■ To configure a shorter relative path by indicating the root folder that holds the modules in the File Cabinet. 	Version 2015 Release 2
paths	Object	optional	<ul style="list-style-type: none"> ■ To create a named alias to a path ■ For testing purposes, pass in an object that serves as a mock-up of another module. ■ To set up a custom name for a SuiteScript module 	Version 2015 Release 2
packages	Object[]	optional	<ul style="list-style-type: none"> ■ To configure a special lookup suitable for traditional CommonJS packages that you want to use as a custom module. 	Version 2015 Release 2
map	Object	optional	<ul style="list-style-type: none"> ■ To specify an alias. ■ To handle multiple names for a module ■ To load a set of identically-named but unique modules, such as dependency on multiple module versions. 	Version 2015 Release 2
config	Object	optional	<ul style="list-style-type: none"> ■ To assign attributes, such as metadata. 	Version 2015 Release 2
shim	Object	optional	<ul style="list-style-type: none"> ■ To prepare a non-AMD JS library for loading. 	Version 2015 Release 2

require.config()

Configuration of a require Object is optional and for advanced usage only. If you must configure a require Object, the @NAmConfig tag is suited for general use and is the preferred way to configure a require Object. However, existing scripts with calls to require.config can use this method with a context argument (although not recommended). Ensure that the call includes a context parameter and that its value is not a file path.

log Object

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

NetSuite loads the log Object is loaded by default for all script types. You do not need to load it manually. However, you can choose to load it via [N/log Module](#), for testing purposes.

log Object Members

- [log.debug\(options\)](#)
- [log.audit\(options\)](#)
- [log.emergency\(options\)](#)
- [log.error\(options\)](#)

For more details about the log Object and its methods, see [N/log Module](#).

util Object

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

NetSuite loads the util Object is loaded by default for all script types. You do not need to load it manually. However, you can choose to load it via [N/util Module](#), for testing purposes.

util Object Members

- [util.isArray\(obj\)](#)
- [util.isBoolean\(obj\)](#)
- [util.isDate\(obj\)](#)
- [utilisFunction\(obj\)](#)
- [util.isNumber\(obj\)](#)
- [utilisObject\(obj\)](#)
- [util.isRegExp\(obj\)](#)
- [util.isString\(obj\)](#)

The util object also includes the following utility methods:

- [util.each\(iterable, callback\)](#)
- [util.extend\(receiver, contributor\)](#)

For more details about the util Object and its methods, see [N/util Module](#).

toString()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to determine an object's type. This is a global method that is loaded by default for all native SuiteScript 2.0 API objects.
	<p> Note: Consider this method a replacement for the instanceof operator (which is not supported). SuiteScript 2.0 members are immutable; you cannot construct or modify a native SuiteScript 2.0 member. Consequently, if you attempt to call instanceof, an undefined error is thrown.</p> <p>For information about <code>toString()</code> as a native JavaScript method, see https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/object/toString.</p>
Returns	The object type as a string
Supported Script Types	All script types
Governance	None
Since	Version 2015 Release 2

Syntax

The following code snippet shows the syntax for this member. It is not a functional example.

```

1 | ...
2 | var type = mapContext.toString();      // When called on mapReduce.MapContext, toString returns "mapReduce.MapContext"
3 | ...

```

JSON object



Note: The content in this help topic pertains to SuiteScript 2.0.

SuiteScript 2.0 supports the JavaScript Object Notation (JSON) standard. You can use the JSON object to parse text as a JSON object and convert strings to JSON notation. For more information, see [JSON.parse\(text\)](#) and [JSON.stringify\(obj\)](#).



Important: The following sections are included as a summary and are intended for reference only. For additional information about JSON, see <http://www.ietf.org/rfc/rfc4627.txt>.

JSON.parse(text)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Parse a string as a JSON object and returns the object. The text parameter must conform to the JSON standard. See http://www.ietf.org/rfc/rfc4627.txt .
Returns	Object
Supported Script Types	All script types
Governance	None
Global object	JSON object
Since	Version 2015 Release 2

Parameters

Parameter	Type	Required / Optional	Description	Since
text	string	Required	Text to parse as a JSON object. The string must conform to the JSON standard.	Version 2015 Release 2
reviver	Function	Optional	Specifies how to transform the parsed value before it is returned	Version 2015 Release 2

Syntax

The following code snippet shows the syntax for this member. It is not a functional example.

```

1 ...
2 var text = '{ "employees" : [ ' +
3     '{ "firstName":"John" , "lastName":"Doe" } , ' +
4     '{ "firstName":"Anna" , "lastName":"Smith" } , ' +
5     '{ "firstName":"Peter" , "lastName":"Jones" } ] }';
6 var obj = JSON.parse(text);
7 var firstEmp = obj.employees[1].firstName + " " + obj.employees[1].lastName;
8 ...

```

JSON.stringify(obj)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Converts a JavaScript object or value to a JSON string For more information about JSON object format, see http://www.ietf.org/rfc/rfc4627.txt .
Returns	JSON string
Supported Script Types	All script types
Governance	None
Global object	JSON object
Since	Version 2015 Release 2

Parameters

Parameter	Type	Required / Optional	Description	Since
value	Object	Required	The value to convert to a JSON string	Version 2015 Release 2
replacer	Function	Optional	Function that changes the behavior of the stringification process	Version 2015 Release 2
space	Object	Optional	A string or number that is used to insert white space in the output JSON string for readability	Version 2015 Release 2

Syntax

The following code snippet shows the syntax for this member. It is not a functional example.

```

1 var contact = {
2     firstName: 'John',
3     lastName : 'Doe',
4     jobTitle : 'CEO'
5 };
6
7 var jsonString = JSON.stringify(contact);

```

This method converts the contact object to the following string:

```
{"firstName": "John", "lastName": "Doe", "jobTitle": "CEO"}
```

Promise Object



Note: The content in this help topic pertains to SuiteScript 2.0.

In SuiteScript 2.0, all client scripts support the use of Promises. With Promises, developers can write asynchronous code that is intuitive and efficient. SuiteScript 2.0 provides promise APIs for selected modules (see [SuiteScript 2.0 Promise APIs](#)). In addition, you can create custom Promises in all client scripts (see [Custom Promises](#)).

A promise is a JavaScript object that represents the eventual result of an asynchronous process. After this object is created, it serves as a placeholder for the future success or failure of the operation. During the period of time that the promise object is waiting, the remaining segments of the script can execute.

A Promise holds one of the following values:

- fulfilled – The operation is successful.
- rejected – The operation failed.
- pending – The operation is still in progress and has not yet been fulfilled or rejected.

When it is first created, a Promise holds the value pending. After the associated process is complete (from success or failure), the value changes to fulfilled or rejected. A success or failure callback function attached to the Promise is called when the process is complete. Note that a Promise can only succeed or fail one time. When the value of the Promise updates to fulfilled or rejected, it cannot change.

For additional information regarding Promises, see <https://www.promisejs.org/>.

SuiteScript 2.0 Promise APIs

SuiteScript 2.0 provides client-side promise APIs. For supported modules members and additional API information, see [SuiteScript 2.0 Modules](#).



Important: Although these modules as a whole are supported in client and server-side scripts, their promise APIs are supported only in client scripts.

The available promise APIs are named so that they correspond with their synchronous counterparts. The distinction is that the promise APIs have names that are suffixed with .promise. For example, the search.create(options) API has a promise version named search.create.promise(options).

The following is a basic example of how to use a promise API in a client script.

```

1 /**
2 * @NAPIVersion 2.0
3 */
4 define(['N/search'],
5   function(search)
6   {
7     function doSomething()
8     {
9       search.create.promise({
10         type: 'salesorder'
11       })
12       .then(function(result) {
13         log.debug("Completed: " + result);
14         //do something after completion
15       })
16       .catch(function(reason) {
17         log.debug("Failed: " + reason)
18         //do something on failure
19       })
20     }
21   }
22 )

```

```

19     });
20   }
21   return
22   {
23     pageInit: doSomething
24   }
25 }
26 );

```

This example demonstrates how to chain promises created with promise APIs.

```

1 /**
2 * @NAPIVersion 2.0
3 */
4 define(['N/search'],
5   function(search)
6   {
7     function doSomething()
8     {
9       var filter = search.createFilter({
10         name: 'mainline',
11         operator: search.Operator.IS,
12         values:[T]
13       });
14       search.create.promise({
15         type: 'salesorder',
16         filters:[filter]
17       })
18     .then(function(searchObj) {
19       return searchObj.run().each.promise(
20         function(result, index){
21           //do something
22         })
23     })
24     .then(function(result) {
25       log.debug("Completed: " + result)
26       //do something after completion
27     })
28     .catch(function(reason) {
29       log.debug("Failed: " + reason)
30       //do something on failure
31     });
32   }
33   return
34   {
35     pageInit: doSomething
36   }
37 }
38 );

```

Custom Promises

The following example shows a custom Promise. Custom Promises do not utilize the SuiteScript 2.0 promise APIs.

```

1 /**
2 * @NAPIVersion 2.0
3 */
4 define(function(){
5   function doSomething(addresses){
6     var promise = new Promise(function(resolve, reject){
7       var url = 'https://your.favorite.maps/api/directions?start=' + addresses.start + '&end=' + addresses.end,
8         isAsync = true,
9         xhr = new XMLHttpRequest();
10
11       xhr.addEventListener('load', function (event) {
12         if (xhr.readyState === 4) {

```

```
13         if (xhr.status === 200) {
14             resolve(xhr.responseText );
15         }
16         else {
17             reject( xhr.statusText );
18         }
19     });
20 );
21 xhr.addEventListener('error', function (event) {
22     reject( xhr.statusText );
23 });
24 xhr.open('GET', url, isAsync);
25 xhr.send();
26 });
27
28 return promise;
29 }
30
31 return {
32     lookupDirections: doSomething
33 };
34});
```

SuiteScript 2.0 Modules

SuiteScript 2.0 APIs are organized into various modules, based on behavior. The following table lists each module and provides a description, the supported script types, and permissions associated with the module. See the help topic [Setting Roles and Permissions for SuiteScript](#) for more information on permissions.

Note: As a best practice, you should load only the modules that are needed by your script. However, you can load all SuiteScript 2.0 or SuiteScript 2.1 modules at one time by passing the modules' parent directory to the define() statement and its callback function: `define(['N'], function(N) { ... })`. This is a convenient way to load all modules, however, doing so will sacrifice the performance advantage of loading only the modules that are needed. We provide this feature so that you can test and familiarize yourself with SuiteScript 2.0 and 2.1. We do not recommend that you load all modules at once in a production environment.

Module	Description	Supported Script Types	Permissions
N/action Module	Load the N/action module to execute business logic to update the state of a record. Action APIs emulate NetSuite user interface buttons.	Client and server scripts	
N/auth Module	Load the N/auth module to change your NetSuite login credentials.	Server scripts	
N/cache Module	Load the N/cache module to enable the caching of needed data and improve performance.	Server scripts	
N/certificateControl Module	Load the N/certificateControl module to enable scripting access to the Digital Certificates list found at Setup > Company > Certificates. You can use this module to find the correct certificate for a subsidiary and check the file type. For more information, see the help topics Digital Signing and Uploading Digital Certificates .	Server scripts	Certificate Management
SuiteScript 2.x APIs	Load modules in the N/commerce namespace to access different assets in the web store context, such as items and shopping cart. The modules within the N/commerce namespace are supported by the latest version of SuiteCommerce and by SuiteCommerce Advanced 2019.2 onwards.	Client and server scripts	
N/compress Module	Load the N/compress module to compress, decompress, and archive files.	Server scripts	
N/config Module	Load the N/config module to access NetSuite configuration settings. See config.Type for a list of supported configuration pages.	Server scripts	
N/crypto Module	Load the N/crypto module to work with hashing, hash-based message authentication (hmac), and symmetrical encryption. You can access a set of wrappers for OpenSSL's hash, hmac, cipher, and decipher methods.	Server scripts	

Module	Description	Supported Script Types	Permissions
N/crypto/certificate Module	Load the N/certificate module to sign XML documents or strings with digital certificates using asymmetric cryptography. In addition to signing XML documents, you can create signer and verifier objects and verify signed documents with this module.	Server scripts	Certificate Access
N/currency Module	Load the N/currency module to work with exchange rates within your NetSuite account. You can use this module to find the exchange rate between two currencies based on a certain date.	Client and server scripts	
N/currentRecord Module	Load the N/currentRecord module to access the record instance that you are currently working on. You can then use the record instance in a client-side context.	Client scripts	
N/email Module	Load the N/email module to send email messages from within NetSuite. You can use the email module to send regular, bulk, and campaign email.	Client and server scripts	
N/encode Module	Load the N/encode module to convert a string to another type of encoding. See encode.Encoding for a list of supported character set encoding.	Server scripts	
N/error Module	Load the N/error module to create your own custom SuiteScript errors. Use these custom errors in try-catch statements to abort script execution.	Server scripts	
N/file Module	Load the N/file module to work with files in NetSuite.	Server scripts	
N/format Module	Load the N/format module to convert strings into a specified format and to parse formatted data into strings.	Client and server scripts	
N/format/i18n Module	Load the N/format/i18n module to format currency.	Client and server scripts	
N/http Module	Load the N/http module to make http calls.	Client and server scripts	
N/https Module	Load the N/https module to make https calls. You can also use this module to encode binary content or securely access a handle to the value in a NetSuite credential field.	Client and server scripts	
N/https/clientCertificate Module	Load the N/clientCertificate module to send SSL requests with a digital certificate.	Server scripts	Certificate Access
N/keyControl Module	Load the N/keyControl module to access key storage.	Server scripts	Key Management
N/log Module	Load the N/log module to access methods for logging script execution details. Module	Client and server scripts	

Module	Description	Supported Script Types	Permissions
	members are also supported by the global log Object .		
N/piremoval Module	Load the N/piremoval module to remove personal information (PI) from system notes, workflow history, and specific field values.	Server scripts	Remove Personal Information Create Remove Personal Information Run
N/plugin Module	Load the N/plugin module to load custom plug-in implementations.	Server scripts	
N/portlet Module	Load the N/portlet module to resize or refresh a form portlet.	Client scripts	
N/query Module	Load the N/query module to create and run searches using the SuiteAnalytics Workbook query engine.	Client and server scripts	SuiteAnalytics Workbook
N/record Module	Load the N/record module to work with NetSuite records.	Client and server scripts	
N/recordContext Module	Load the N/recordContext module to get the context type of a record.	Client and server scripts	
N/redirect Module	Load the N/redirect module to redirect users to a URL, a Suitelet, a record, a task link, a saved search, or an unsaved search.	Server scripts	
N/render Module	Load the N/render module to create forms or email from templates and to print to PDF or HTML.	Server scripts	Advanced PDF/HTML Templates Custom PDF Layouts
N/runtime Module	Load the N/runtime module to access runtime settings for company, script, session, system, user, or version.	Client and server scripts	
N/search Module	Load the N/search module to create and run on-demand or saved searches and analyze and iterate through the search results. You can search for a single record by keywords, create saved searches, search for duplicate records, or return a set of records that match filters you define.	Client and server scripts	Perform Search Persist Search Publish Search
N/sftp Module	Load the N/sftp module to connect to a remote FTP server via SFTP and transfer files.	Server scripts	Key Access (needed when public key authentication is used)
N/sso Module	Load the N/sso module to generate outbound single sign-on (SuiteSignOn) tokens	Client and server scripts	
N/task Module	Load the N/task module to create tasks and place them in the internal NetSuite scheduling or task queue. Use this module to schedule scripts, run Map/Reduce scripts, import CSV files, merge duplicate records, and execute asynchronous workflows.	Server scripts	Tasks

Module	Description	Supported Script Types	Permissions
N/task/accounting/recognition Module	Load the task/accounting/recognition module to merge revenue arrangements or revenue elements. This module lets you combine revenue arrangements or revenue elements from multiple sources to represent a single contract for revenue allocation and recognition.	Server scripts	(Transactions) Revenue Arrangement
N/transaction Module	Load the N/transaction module to void transactions.	Client and server scripts	
N/translation Module	Load the N/translation module to load NetSuite Translation Collections in SuiteScript.	Client and server scripts	
N/ui/dialog Module	Load the N/dialog module to create a modal dialog that is displayed until a button on the dialog is pressed.	Client scripts	
N/ui/message Module	Load the N/message module to display a message at the top of the screen under the menu bar.	Client scripts	
N/ui/serverWidget Module	Load the N/serverWidget module to work with the user interface within NetSuite.	Server scripts	
N/url Module	Load the N/url module to determine URL navigation paths within NetSuite or format URL strings.	Client and server scripts	
N/util Module	Load the N/util module to manually access util methods. Module members are also supported by the global util Object .	Client and server scripts	
N/workflow Module	Load the N/workflow module to initiate new workflow instances or trigger existing workflow instances.	Server scripts	Workflow
N/xml Module	Load the N/xml module to validate, parse, read, and modify XML documents.	Client and server scripts	

N/action Module

 **Note:** This content in this topic applies to SuiteScript 2.0.

The N/action module APIs let you execute business logic to update the state of records in view mode. To execute business logic on records that you are editing, use the record macro APIs, which are included in the [N/record Module](#) module. See [Record Object Members](#) and [Macro Object Members](#). Action and Macro APIs are the programmatic equivalent to clicking a button in the UI. To learn more, see the help topic [Overview of Record Action and Macro APIs](#).

The changes that you make to records with N/action module APIs are persisted in the database immediately. For example, consider the timebill record. After you click the **Approve** button in the UI, the timebill and its entries are saved in an approved state, and this change is immediately updated in the database.

Governance for action module APIs varies for actions and record types. See the action help for governance information specific to actions and record types.

A limited number of individual actions for specific record types are supported. For details, see the help topic [Supported Record Actions](#).



Note: For supported script types, see individual member topics listed below.

- N/action Module Members
- Action Object Members
- N/action Module Script Samples

N/action Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	action.Action	Object	Client and server-side scripts	Encapsulates a NetSuite record action.
	Plain JavaScript Object	Object	Client and server-side scripts	A plain JavaScript object of actions available for a record type.
Method	action.execute(options)	Object	Client and server-side scripts	Executes the record action and returns action results in an object.
	action.execute.promise(options)	Promise	Client scripts	Asynchronously executes the record action and returns the action results in an object.
	action.executeBulk(options)	string	Client and server-side scripts	Executes an asynchronous bulk record action and returns its task ID for later status inquiry.
	action.find(options)	Object	Client and server-side scripts	Returns a plain JavaScript object of available record actions for the given record type.
	action.find.promise(options)	Promise	Client scripts	Asynchronously returns a plain JavaScript object of available record actions for the given record type.
	action.get(options)	action.Action	Client and server-side scripts	Returns an executable record action for the given record type.
	action.get.promise(options)	Promise	Client scripts	Asynchronously returns an executable record action for the given record type.

Action Object Members

The following members are called on [action.Action](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Action(options)	Object	Client and server-side scripts	Executes the action and returns the action results in an object.

Member Type	Name	Return Type/ Value Type	Supported Script Types	Description
	Action.promise(options)	Promise	Client scripts	Executes the action asynchronously and returns the action results in an object.
	Action.execute(options)	Object	Client and server-side scripts	Executes the action and returns the action results in an object.
	Action.execute.promise(options)	Promise	Client scripts	Executes the action asynchronously and returns the action results in an object.
	Action.executeBulk(options)	string	Client and server-side scripts	Executes an asynchronous bulk record action and returns its task ID for later status inquiry.
	action.getBulkStatus(options)	Object	Client and server-side scripts	Returns the current status of action.executeBulk(options) with the given task ID.
Property	Action.description	string	Client and server-side scripts	The action description.
	Action.id	string	Client and server-side scripts	The ID of the action. For a list of action IDs, see the help topic Supported Record Actions .
	Action.label	string	Client and server-side scripts	The action label.
	Action.parameters	Object	Client and server-side scripts	The action parameters.
	Action.recordType	string	Client and server-side scripts	The type of the record on which the action is to be performed. For a list of record types, see record.Type .

N/action Module Script Samples

These samples use the `require` function, so that you can copy each script into the debugger and test it. Keep in mind that you must use the `define` function in your entry point script (the script you attach to a script record). For more information, see [SuiteScript 2.0 Global Objects](#) and [SuiteScript 2.0 Script Types](#).



Important: The samples included in this section are intended to show how actions work in SuiteScript at a high-level. For specific samples, see the help topic [Supported Record Actions](#).



Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following server script sample finds and executes an action on the timebill record without promises.

```

1 /**
2  * @NApiVersion 2.x

```

```

3  /*
4
5 require(['N/action', 'N/record'], function(action, record) {
6   // create timebill record
7   var rec = record.create({type: 'timebill', isDynamic: true});
8   rec.setValue({fieldId: 'employee', value: 104});
9   rec.setValue({fieldId: 'location', value: 312});
10  rec.setValue({fieldId: 'hours', value: 5});
11  var recordId = rec.save();
12
13  var actions = action.find({
14    recordType: 'timebill',
15    recordId: recordId
16
17  });
18
19  log.debug("We've got the following actions: " + Object.keys(actions));
20  if (actions.approve) {
21    var result = actions.approve();
22    log.debug("Timebill has been successfully approved");
23  } else {
24    log.debug("The timebill is already approved");
25  }
26});
27
28 // Outputs the following:
29 // We've got the following actions: approve, reject
30 // Timebill has been successfully approved

```

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following client script sample asynchronously finds actions available for a timebill record and then executes one with promises.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType ClientScript
4 */
5
6 require(['N/action', 'N/record'], function(action, record) {
7   // create timebill record
8   var rec = record.create({type: 'timebill', isDynamic: true});
9   rec.setValue({fieldId: 'employee', value: 104});
10  rec.setValue({fieldId: 'location', value: 312});
11  rec.setValue({fieldId: 'hours', value: 5});
12  var recordId = rec.save();
13
14 // find all qualified actions and then execute approve if available
15  action.find.promise({
16    recordType: 'timebill',
17    recordId: recordId
18  }).then(function(actions) {
19    console.log("We've got the following actions: " + Object.keys(actions));
20    if (actions.approve) {
21      actions.approve.promise().then(function(result) {
22        console.log("Timebill has been successfully approved");
23      });
24    } else {
25      console.log("The timebill is already approved");
26    }
27  });
28});
29
30 // Outputs the following:
31 // We've got the following actions:
32 // The timebill has been successfully approved

```



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to use the action.executeBulk(options).

```

1  /**
2  * @NApiVersion 2.x
3  */
4
5  require(['N/action', 'N/util']) function(action, util) {
6
7      // 1a) Bulk execute the specified action on a provided list of record IDs.
8      // The params property is an array of parameter objects where each object contains mandatory recordId and arbitrary additional parameters.
9      var handle = action.executeBulk({
10          recordType: "timebill",
11          id: "approve",
12          params: [{ recordId: 1, note: "this is a note for 1" },
13                  { recordId: 5, note: "this is a note for 5" },
14                  { recordId: 23, note: "this is a note for 23" }]
15      });
16  });
17
18      // 1b) Bulk execute the specified action on a provided list of record IDs.
19      // The parameters in the previous sample are very similar and can be generated programatically using the map function.
20      var searchResults = /* result of a search, e.g. [1, 5, 23] */;
21      var handle = action.executeBulk({
22          recordType: "timebill",
23          id: "approve",
24          params: searchResults.map(function(v) {
25              return { recordId: v, note: "this is a note for " + v };
26          })
27      });
28
29      // 2a) Bulk execute the specified action on a provided list of record IDs.
30      // This time with homogenous parameters, i.e. all parameter objects are equal except recordId.
31      var handle = action.executeBulk({
32          recordType: "timebill",
33          id: "approve",
34          params: searchResults.map(function(v) {
35              return { recordId: v, foo: "bar", name: "John Doe" };
36          })
37      });
38
39      // 2b) Bulk execute the specified action on a provided list of record IDs.
40      // This time with homogenous parameters. Equivalent to the previous sample.
41      var commonParams = {foo: "bar", name: "John Doe"};
42      var handle = action.executeBulk({
43          recordType: "timebill",
44          id: "approve",
45          params: searchResults.map(function(v) {
46              return util.extend({recordId: v}, commonParams);
47          })
48      });
49
50      // 3) Bulk execute the specified action on a provided list of record IDs.
51      // This is the simplest usage with no extra parameters besides the record ID.
52      var handle = action.executeBulk({
53          recordType: "timebill",
54          id: "approve",
55          params: searchResults.map(function(v) {return {recordId: v}})
56      });
57
58      // 4) Bulk execute the specified action on all record instances that qualify.
59      // Since we don't have a list of recordIds in hand, we only provide the callback
60      // that will later be used to transform a recordId to the corresponding parameters object.
61      var handle = action.executeBulk({
62          recordType: "timebill",
63          id: "approve",

```

```

64     condition: action.ALL_QUALIFIED_INSTANCES,
65     paramCallback: function(v) {
66       return { recordId: v, note: "this is a note for " + v };
67     }
68   });
69
70 // 5) Get a particular action for a particular record type.
71 var approveTimebill = action.get({
72   recordType: "timebill",
73   id: "approve"
74 });
75
76 // 6) Bulk execute the previously obtained action on a provided list of record IDs.
77 // Params are generated the same way as above in action.executeBulk().
78 var handle = approveTimebill.executeBulk({
79   params: searchResults.map(function(v) {
80     return { recordId: v, note: "this is a note for " + v };
81   })
82 });
83
84 // 7) Bulk execute the previously obtained action on all record instances that qualify.
85 var handle = approveTimebill.executeBulk({
86   condition: action.ALL_QUALIFIED_INSTANCES,
87   paramCallback: function(v) {
88     return { recordId: v, note: "this is a note for " + v };
89   }
90 });
91
92 // 8) Get status of a bulk action execution.
93 var res = action.getBulkStatus({taskId: handle}); // returns a RecordActionTaskStatus object
94 log.debug(res.status);
95 });

```

action.Action



Note: This content in this topic applies to SuiteScript 2.0.

Object Description	Encapsulates a NetSuite record action. This object is returned by the action.get(options) and action.find(options) methods.
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/action Module
Methods and Properties	Action Object Members
Since	2018.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var action = actionMod.get({recordType: 'timebill', id: 'approve'});
4 ...
5 // Add additional code

```

Action(options)



Note: This content in this topic applies to SuiteScript 2.0.

Method Description	<p>Executes the action and returns the action result in a plain JavaScript object.</p> <p>The action result is returned in an object. The <code>response</code> property of the results object shows the action result. If the action fails, it is listed in the results object's <code>notifications</code> property. If the action executes successfully, the <code>notifications</code> property is usually empty.</p> <p>If the Action object is qualified (it is a result of an <code>action.get()</code> or <code>action.find()</code> call that provides the <code>recordId</code>), then it is not required to provide a <code>recordId</code> and the <code>options.params.recordId</code> parameter is optional. If <code>options.params.recordId</code> is provided during execution, it takes precedence over the <code>recordId</code> stored in the Action object.</p>
	<p> Note: Replace Action with the name of the action you are executing.</p>
Returns	Object
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Parent Object	<code>action.Action</code>
Sibling Object Members	Action Object Members
Since	2018.2

Parameters



Note: The parameters that are required vary for action types. The only parameter that is always required is `options.recordId`, unless the action object is qualified. An action object is qualified if it is the result of an `action.get()` or `action.find()` call that provides the `recordId`.

Parameter	Type	Required / Optional	Description
<code>options.Object</code>	Object	required or optional	The parameters that need to be provided depend on the action implementation. See the action help.
<code>options.params.recordId</code>	string	required or optional	<p>The record instance ID of the record on which the action is to be performed.</p> <p>This is the NetSuite record internal ID.</p>

Errors

Syntax

Error Code	Thrown If
<code>SSS_MISSING_REQD_ARGUMENT</code>	A required parameter is missing.



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var result = action({recordId: 1});
4 ...
5 // Add additional code

```

Action.promise(options)



Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Executes the action asynchronously and returns the action result in a plain JavaScript object. The action result is returned in an object. The <code>response</code> property of the results object shows the action result. If the action fails, it is listed in the results object's <code>notifications</code> property. If the action executes successfully, the <code>notifications</code> property is usually empty. If the Action object is qualified (it is a result of an <code>action.get()</code> or <code>action.find()</code> call that provides the <code>recordId</code>), then it is not required to provide a <code>recordId</code> and the <code>options.params.recordId</code> parameter is optional. If <code>options.params.recordId</code> is provided during execution, it takes precedence over the <code>recordId</code> stored in the Action object.
	Note: Replace Action with the name of the action you are executing.
	Note: The parameters and errors thrown for this method are the same as those for Action(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Parent Object	action.Action
Sibling Object Members	Action Object Members
Since	2018.2

Parameters



Note: The parameters that are required vary for action types. The only parameter that is always required is `options.recordid`, unless the action object is qualified. An action object is qualified if it is the result of an `action.get()` or `action.find()` call that provides the `recordId`.

Parameter	Type	Required / Optional	Description
<code>options.Object</code>	Object	required or optional	The parameters that need to be provided depend on the action implementation. See the action help.

Parameter	Type	Required / Optional	Description
options.params.recordId	string	required or optional	The record instance ID of the record on which the action is to be performed. This is the NetSuite record internal ID.

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 action.promise({recordId: 1}).then(function(result) { /* process result here */ });
4 ...
5 // Add additional code

```

Action.execute(options)



Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Executes the action and returns the action result in a object. The response property of the result object holds the actual response returned by the action implementation. The notifications property of the result object is an array of notification objects. It contains the details of errors and warnings that occurred during action execution. If the action executes successfully, the notifications property is usually empty. If the Action object is qualified (it is a result of an action.get() or action.find() call that provides the recordId), then it is not required to provide a recordId and the options.params.recordId parameter is optional. If options.params.recordId is provided during execution, it takes precedence over the recordId stored in the Action object.
Returns	Object
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Parent Object	action.Action
Sibling Object Members	Action Object Members
Since	2018.2

Parameters

i Note: The parameters that are required vary for action types. The only parameter that is always required is options.recordId, unless the action object is qualified. An action object is qualified if it is the result of an action.get() or action.find() call that provides the recordId.

Parameter	Type	Required / Optional	Description
options.object	Object	required or optional	The parameters that need to be provided depend on the action implementation. See the action help.
options.params.recordId	string	required or optional	The record instance ID of the record on which the action is to be performed. This is the NetSuite record internal ID.

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var result = action.execute({recordId: 1});
4 ...
5 // Add additional code

```

Action.execute.promise(options)

i Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Executes the action asynchronously and returns the action result in a plain JavaScript object. The action result is returned in an object. The response property of the results object shows the action result. If the action fails, it is listed in the results object's notifications property. If the action executes successfully, the notifications property is usually empty. If the Action object is qualified (it is a result of an action.get() or action.find() call that provides the recordId), then it is not required to provide a recordId and the options.params.recordId parameter is optional. If options.params.recordId is provided during execution, it takes precedence over the recordId stored in the Action object.
Returns	Promise Object

Supported Script Types	Client scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Parent Object	action.Action
Sibling Object Members	Action Object Members
Since	2018.2

Parameters

Note: The parameters that are required vary for action types. The only parameter that is always required is options.recordId, unless the action object is qualified. An action object is qualified if it is the result of an action.get() or action.find() call that provides the recordId.

Parameter	Type	Required / Optional	Description
options.Object	Object	required or optional	The parameters that need to be provided depend on the action implementation. See the action help.
options.params.recordId	string	required or optional	The record instance ID of the record on which the action is to be performed. This is the NetSuite record internal ID.

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 action.execute.promise({recordId: 1}).then(function(result) { /* process result here */ });
4 ...
5 // Add additional code

```

Action.executeBulk(options)

Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Executes an asynchronous bulk record action and returns its task ID for status queries with action.getBulkStatus(options) .
---------------------------	---

Returns	string
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	50 units
Module	N/action Module
Parent Object	action.Action
Sibling Object Members	Action Object Members
Since	2019.1

Parameters

i Note: The options.params array consists of parameter objects. The values that are required in each parameter object vary for action types. The only value that is always required is options.recordId, unless the action object is qualified. An action object is qualified if it is the result of an action.get() or action.find() call that provides the recordId.

Parameter	Type	Required / Optional	Description
options.params	array	optional	<p>The options.params parameter is mutually exclusive to options.condition and options.paramCallback.</p> <p>An array of parameter objects. Each object corresponds to one record ID of the record for which the action is to be executed. The object has the following form:</p> <pre>1 {recordId: 1, someParam: 'example1', otherParam: 'example2'}</pre>
options.condition	string	optional	<p>The condition used to select record IDs of records for which the action is to be executed. Only the action.ALL_QUALIFIED_INSTANCES constant is currently supported.</p> <p>The action.ALL_QUALIFIED_INSTANCES condition only works correctly if the author of the record action has implemented the findInstances method of the RecordActionQualifier interface. An example of such action is approve on the timebill and timesheet records.</p>
options.paramCallback	string	optional	the name of the function that takes a record ID and returns the parameter object for the specified record ID.

Errors

Error Code	Thrown If
SSS_INVALID_RECORD_TYPE	The specified record type is invalid.
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.
SSS_INVALID_ACTION_ID	The specified action does not exist on the specified record type. – or –

Error Code	Thrown If
	The action exists, but cannot be executed on the specified record instance.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2
3 var actionObj = action.get({
4     recordType: 'timebill',
5     id: 'approve'
6 });
7
8 var handle = actionObj.executeBulk({
9     params: [
10         {
11             recordId: 1, note: 'this is a note for 1'
12         },
13         {
14             recordId: 5, note: 'this is a note for 5'
15         },
16         {
17             recordId: 23, note: 'this is a note for 23'
18         }
19     ]
20 });
21 ...
22 // Add additional code

```

action.getBulkStatus(options)

Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Returns the current status of action.executeBulk(options) for the specified task ID. The bulk execution status is returned in a status object.
Returns	RecordActionTaskStatus Object Members
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Sibling Object Members	N/action Module Members
Since	2019.1

Parameters

Parameter	Type	Required / Optional	Description
options.taskId	string	required	The task ID returned by a previous action.executeBulk(options) call.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2
3 // Obtain the status as a RecordActionTaskStatus object
4 ...
5 var res = action.getBulkStatus({
6     taskId: handle
7 });
8 // Add additional code

```

Action.description



Note: This content in this topic applies to SuiteScript 2.0.

Property Description	The action description.
Type	string
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/action Module
Sibling Object Members	Action Object Members
Since	2018.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var description = action.description; // get the action description
4 ...
5 // Add additional code

```

Action.id



Note: This content in this topic applies to SuiteScript 2.0.

Property Description	The ID of the action. For a list of action IDs, see the help topic Supported Record Actions .
Type	string
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/action Module

Sibling Object Members	Action Object Members
Since	2018.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var id = action.id; // get the id of the action
4 ...
5 // Add additional code

```

Action.label



Note: This content in this topic applies to SuiteScript 2.0.

Property Description	The action label.
Type	string
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/action Module
Sibling Object Members	Action Object Members
Since	2018.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var label = action.label; // get the action label
4 ...
5 // Add additional code

```

Action.parameters



Note: This content in this topic applies to SuiteScript 2.0.

Property Description	The action parameters.
Type	Object
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .

Module	N/action Module
Sibling Object Members	Action Object Members
Since	2018.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var params = action.parameters; // get the action parameters
4 ...
5 // Add additional code

```

Action.recordType

i Note: This content in this topic applies to SuiteScript 2.0.

Property Description	The type of the record on which the action is to be performed. For a list of record types, see record.Type .
Type	string
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/action Module
Sibling Object Members	Action Object Members
Since	2018.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var recordType = action.recordType; // get the record type
4 ...
5 // Add additional code

```

action.execute(options)

i Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Executes the record action and returns the action results in a plain JavaScript object. If the action fails, it is listed in the results object's notifications property. If the action executes successfully, the notifications property is usually empty.
---------------------------	--

Returns	Object
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Sibling Object Members	N/action Module Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.recordType	string	required	The record type. For a list of record types, see record.Type .
options.id	string	required	The action ID. For a list of action IDs, see the help topic Supported Record Actions .
options.params	Object	required	Action arguments.
options.params.recordId	string	required	The record instance ID. This is the NetSuite record internal ID.

Errors

Error Code	Thrown If
SSS_INVALID_RECORD_TYPE	The specified record type is invalid.
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.
SSS_INVALID_ACTION_ID	The specified action does not exist on the specified record type. – or – The action exists, but cannot be executed on the specified record instance.
RECORD_DOES_NOT_EXIST	The specified record instance does not exist.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#) and [Revenue Arrangement Record Actions](#).

```

1 // Add additional code
2 ...
3 var result = actionMod.execute({id: 'note', recordType: 'timebill', params: {recordId:1}}
```

```

4 });
5 ...
6 // Add additional code

```

action.execute.promise(options)

Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Executes the record action asynchronously. If the action fails, it is listed in the results object's notifications property. If the action executes successfully, the notifications property is usually empty.
	Note: The parameters and errors thrown for this method are the same as those for action.execute(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Sibling Object Members	N/action Module Members
Since	2018.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.recordType	string	required	The record type. For a list of record types, see record.Type .
options.id	string	required	The action ID. For a list of action IDs, see the help topic Supported Record Actions .
options.params	Object	required	Action arguments.
options.params.recordId	string	required	The record instance ID. This is the NetSuite record internal ID.

Errors

Error Code	Thrown If
SSS_INVALID_RECORD_TYPE	The specified record type is invalid.

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.
SSS_INVALID_ACTION_ID	The specified action does not exist on the specified record type. – or – The action exists, but cannot be executed on the specified record instance.
RECORD_DOES_NOT_EXIST	The specified record instance does not exist.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 actionMod.execute.promise({id: 'note', recordType: 'timebill', params: {recordId: 1}}).then(function(result) {
4     // do something with the result
5 });
6 ...
7 // Add additional code

```

action.executeBulk(options)



Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Executes an asynchronous bulk record action and returns its task ID for status queries with action.getBulkStatus(options) . The options.params parameter is mutually exclusive to options.condition and options.paramCallback.
Returns	string
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	50 units
Module	N/action Module
Sibling Object Members	N/action Module Members
Since	2019.1

Parameters



Note: The options parameter is a JavaScript object. The options.params array consists of parameter objects. The values that are required in each parameter object vary for action types. The only value that is always required is recordId.

Parameter	Type	Required / Optional	Description
options.recordType	string	required	The record type.

Parameter	Type	Required / Optional	Description
			For a list of record types, see record.Type .
options.id	string	required	The action ID.
options.params	array	optional	An array of parameter objects. Each object corresponds to one record ID of the record for which the action is to be executed. The object has the following form: 1 {recordId: 1, someParam: 'example1', otherParam: 'example2'} The recordId parameter is always mandatory, other parameters are optional and are specific to the particular action.
options.condition	string	optional	The condition used to select record IDs of records for which the action is to be executed. Only the action.ALL_QUALIFIED_INSTANCES constant is currently supported. The action.ALL_QUALIFIED_INSTANCES condition only works correctly if the author of the record action has implemented the findInstances method of the RecordActionQualifier interface. An example of such action is approve on the timebill and timesheet records.
options.paramCallback	string	optional	Function that takes record ID and returns the parameter object for the specified record ID.

Errors

Error Code	Thrown If
SSS_INVALID_RECORD_TYPE	The specified record type is invalid.
SSS_MISSING_REQD_ARGUMENT	The options.recordType parameter is missing or undefined.
SSS_INVALID_ACTION_ID	The specified action does not exist on the specified record type. - or - The action exists, but cannot be executed on the specified record instance.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var handle = action.executeBulk({
4     recordType: 'timebill',
5     id: 'approve',
6     params: [{ recordId: 1, note: 'this is a note for 1' },
7               { recordId: 5, note: 'this is a note for 5' },
8               { recordId: 23, note: 'this is a note for 23' }]
9 }
10 });
11 // Add additional code

```

action.find(options)



Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Performs a search for available record actions. If only the <code>recordType</code> parameter is specified, all actions available for the record type are returned. If the <code>recordId</code> parameter is also specified, then only actions that qualify for execution on the given record instance are returned. If the <code>id</code> parameter is specified, then only the action with the specified action ID is returned. This method returns a plain JavaScript object of NetSuite record actions available for the record type. The object contains one or more <code>action.Action</code> objects. If there are no available actions for the specified record type, an empty object is returned. If the <code>recordId</code> is specified in this call, the actions that are found are considered qualified. You do not have to provide the <code>recordId</code> to execute a qualified action.
Returns	Object
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Sibling Object Members	N/action Module Members
Since	2018.2

Parameters



Note: The `options` parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.recordType</code>	string	required	The record type. For a list of record types, see record.Type .
<code>options.recordId</code>	string	optional	The record instance ID.
<code>options.id</code>	string	optional	The action ID.

Errors

Error Code	Thrown If
<code>SSS_INVALID_RECORD_TYPE</code>	The specified record type is invalid.
<code>SSS_MISSING_REQD_ARGUMENT</code>	The <code>options.recordType</code> parameter is missing or undefined.
<code>SSS_INVALID_ACTION_ID</code>	The specified action does not exist on the specified record type.

Error Code	Thrown If
	- or - The action exists, but cannot be executed on the specified record instance.
RECORD_DOES_NOT_EXIST	The specified record ID does not exist.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var actions = action.find({
4     recordType: 'timebill',
5     recordId: recordId
6 });
7 ...
8 // Add additional code

```

action.find.promise(options)



Note: This content in this topic applies to SuiteScript 2.0.

Method Description	<p>Performs a search for available record actions asynchronously. If only the <code>recordType</code> parameter is specified, all actions available for the record type are returned. If the <code>recordId</code> parameter is also specified, then only actions that qualify for execution on the given record instance are returned. If the <code>id</code> parameter is specified, the only the action with the specified action ID is returned.</p> <p>This method returns a plain JavaScript object of NetSuite record actions available for the record type. The object contains one or more <code>action.Action</code> objects. If there are no available actions for the specified record type, an empty object is returned.</p> <p>If the <code>recordId</code> is specified in this call, the actions that are found are considered qualified. You do not have to provide the <code>recordId</code> to execute a qualified action.</p> <p>Note: The parameters and errors thrown for this method are the same as those for <code>action.find(options)</code>. For more information on promises, see Promise Object.</p>
Returns	Promise Object
Supported Script Types	<p>Client scripts</p> <p>For additional information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/action Module
Sibling Object Members	N/action Module Members
Since	2018.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.recordType	string	required	The record type. For a list of record types, see record.Type .
options.recordId	string	optional	The record instance ID.
options.id	string	optional	The action ID.

Errors

Error Code	Thrown If
SSS_INVALID_RECORD_TYPE	The specified record type is invalid.
SSS_MISSING_REQD_ARGUMENT	The options.recordType parameter is missing or undefined.
SSS_INVALID_ACTION_ID	The specified action does not exist on the specified record type. – or – The action exists, but cannot be executed on the specified record instance.
RECORD_DOES_NOT_EXIST	The specified record ID does not exist.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var promise = action.find.promise({recordType: 'timebill'});
4     promise.then(function(actionList) {
5         // do something with the list of actions
6     });
7 ...
8 // Add additional code

```

action.get(options)



Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Returns an executable record action for the specified record type. If the recordId parameter is specified, the action object is returned only if the specified action can be executed on the specified record instance.
Returns	<code>action.Action</code>

Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Sibling Object Members	N/action Module Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.recordType	string	required	The record type. For a list of record types, see record.Type .
options.recordId	string	optional	The record instance ID.
options.id	string	required	The ID of the action. For a list of action IDs, see the help topic Supported Record Actions .

Errors

Error Code	Thrown If
SSS_INVALID_RECORD_TYPE	The specified record type is invalid.
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.
SSS_INVALID_ACTION_ID	The specified action does not exist on the specified record type. – or – The action exists, but cannot be executed on the specified record instance.
RECORD_DOES_NOT_EXIST	The specified record instance does not exist.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/action Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var action = actionMod.get({recordType: 'timebill', id: 'approve'});
4 ...
5 // Add additional code

```

action.get.promise(options)



Note: This content in this topic applies to SuiteScript 2.0.

Method Description	Returns an executable record action for the specified record type asynchronously. If the recordId parameter is specified, the action object is returned only if the specified action can be executed on the specified record instance. Note: The parameters and errors thrown for this method are the same as those for action.get(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/action Module
Sibling Object Members	N/action Module Members
Since	2018.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.recordType	string	required	The record type. For a list of record types, see record.Type .
options.recordId	string	optional	The record instance ID.
options.id	string	required	The ID of the action. For a list of action IDs, see the help topic Supported Record Actions .

Errors

Error Code	Thrown If
SSS_INVALID_RECORD_TYPE	The specified record type is invalid.
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.
SSS_INVALID_ACTION_ID	The specified action does not exist on the specified record type. - or - The action exists, but cannot be executed on the specified record instance.

Error Code	Thrown If
RECORD_DOES_NOT_EXIST	The specified record instance does not exist.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 actionMod.get.promise({recordType: 'timebill', id: 'approve'}).then(function(action) {
4     // do something with the action object
5 });
6 ...
7 // Add additional code

```

N/auth Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/auth module when you want to change your NetSuite login credentials.

- [N/auth Module Members](#)
- [N/auth Module Script Sample](#)

N/auth Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	auth.changeEmail(options)	void	Server scripts	Changes the current user's NetSuite email address (user name).
	auth.changePassword(options)	void	Server scripts	Changes the current user's NetSuite password.

N/auth Module Script Sample



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to change the currently logged-in user's NetSuite email address and password.



Warning: When you run this sample code in the SuiteScript Debugger, it logs an actual request to change the email and then changes the password.



Important: The value used in this sample for the password and email fields are placeholders. Before using this sample, replace the password and email field values with valid values from your NetSuite account. If you run a script with an invalid value, an error may occur.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 //The following script changes the currently logged-in user's NetSuite email address and password.
6 require(['N/auth'],function(auth) {
7     function changeEmailAndPassword() {
8         var password = 'myCurrentPassword';
9         auth.changeEmail({
10             password: password,
11             newEmail: 'auth_test@newemail.com'
12         });
13         auth.changePassword({
14             currentPassword: password,
15             newPassword: 'myNewPa55Word'
16         });
17     }
18     changeEmailAndPassword();
19 });

```

auth.changeEmail(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Changes the current user's NetSuite email address (user name).
Returns	void
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/auth Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.newEmail	string	required	The logged in user's NetSuite new email address.
options.password	string	required	The logged in user's current NetSuite password.

Parameter	Type	Required / Optional	Description
options.onlyThisAccount	boolean	optional	<p>Determines which accounts the email address is changed for. If set to true, the email address change is applied only to roles within the current account. If set to false, the email address change is applied to all accounts and roles.</p> <p>The default value is true.</p>

Errors

Error Code	Thrown If
INVALID_EMAIL	The options.newEmail is invalid.
INVALID_PSWD	<p>The options.password does not conform to the password rules.</p> <p>See the help topic Creating a Strong Password for information on valid passwords.</p>
SSS_MISSING_REQD_PARAMETER	The options.newEmail or options.password parameter is not specified.
WRONG_PARAMETER_TYPE	The options.onlyThisAccount is not specified as a boolean value.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/auth Module Script Sample](#).

```

1 //Add additional code
2 ...
3 auth.changeEmail({
4   password: 'mycurrentPWD',
5   newEmail: 'jwolf@netsuite.com',
6   onlyThisAccount: true
7 });
8 ...
9 //Add additional code

```

auth.changePassword(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Changes the current user's NetSuite password. See the help topic Creating a Strong Password for information on valid passwords.
Returns	void
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/auth Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.currentPassword	string	required	The logged in user's current NetSuite password
options.newPassword	string	required	The logged in user's new NetSuite password. See the help topic Creating a Strong Password for information on valid passwords.

Errors

Error Code	Thrown If
INVALID_PSWD	The options.password does not conform to the password rules. See the help topic Creating a Strong Password for information on valid passwords.
INVALID_EMAIL	The options.newEmail is invalid.
SSS_MISSING_REQD_ARGUMENT	The options.currentPassword or options.newPassword is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/auth Module Script Sample](#).

```

1 //Add additional code
2 ...
3 auth.changePassword({
4   currentPassword: 'mycurrentPWD',
5   newPassword: 'mynewPWD_2'
6 });
7 ...
8 //Add additional code

```

N/cache Module

Note: The content in this help topic pertains to SuiteScript 2.0.

Load the cache module to enable temporary, short-term storage of data. Data is stored in the cache according to its specified time to live, or ttl. The ttl is specified in the [Cache.put\(options\)](#) method options.ttl parameter. The cache module is supported by all server-side script types.

Using a cache improves performance by eliminating the need for scripts in your account to retrieve the same piece of data more than one time. You can create a cache that is accessible at any of three levels: A

cache can be available (1) to the current script only, (2) to all server-side scripts in the current bundle, or (3) to all server-side scripts in your NetSuite account.

- [N/cache Module Members](#)
- [Cache Object Members](#)
- [N/cache Module Script Sample](#)

N/cache Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	cache.Cache	Object	Server-side scripts	Encapsulates a segment of memory that can be used to temporarily store data on a short-term basis.
Method	cache.getCache(options)	cache.Cache	Server-side scripts	Checks for a cache object with the specified name. If the cache exists, this method returns the cache object. If the cache does not exist, the system creates it.
Enum	cache.Scope	enum	Server-side scripts	An enum used to populate the Cache.scope property.

Cache Object Members

The following members are called on [cache.Cache](#).

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	Cache.get(options)	string	Server-side scripts	Retrieves a value from the cache based on a key that you provide. If the requested value is not present or no longer in the cache, the method calls the user-defined function identified by the method's options.loader parameter. If the value provided by that function is not a string, the system uses JSON.stringify() to convert it. The string value is then cached and returned.
	Cache.put(options)	string	Server-side scripts	Puts a value into the cache. If the value provided is not a string, the system uses JSON.stringify() to convert the value to a string. This data is not persistent.
	Cache.remove(options)	string	Server-side scripts	Removes a value from the cache.
Property	Cache.name	string	Server-side scripts	A label that identifies the cache.
	Cache.scope	string	Server-side scripts	A value that describes the availability of the cache. A cache can be made available to the current script only, to

Member Type	Name	Return Type/ Value Type	Supported Script Types	Description
				all scripts in the current bundle, or to all scripts in your NetSuite account.

N/cache Module Script Sample

i Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to retrieve the name of a city based on a ZIP code using a Suitelet. To speed up processing, the Suitelet uses a cache. In this sample, ZIP code is the key used to retrieve city names from the cache. For any ZIP code provided, if the corresponding city value is not already stored in the cache, a loader function is called. This loader function is a custom module called zipCodeDatabaseLoader (shown in the second script sample). It loads a CSV file and uses it to find the requested value.

i Note: This sample depends on a CSV file that must exist before the script is run. The sample CSV file is available [here](#).

```

1  /**
2   * @NApiVersion 2.x
3   * @NScriptType Suitelet
4   */
5 //This script retrieves the name of a city based on a ZIP code, using a cache.
6 define(['N/cache', '/SuiteScripts/zipToCityIndexCacheLoader'], function(cache, lib) {
7     const ZIP_CODES_CACHE_NAME = 'ZIP_CODES_CACHE';
8     const ZIP_TO_CITY_IDX_JSON = 'ZIP_TO_CITY_IDX_JSON';
9
10    function getZipCodeToCityLookupObj() {
11        var zipCache = cache.getCache({
12            name: ZIP_CODES_CACHE_NAME
13        });
14        var zipCacheJson = zipCache.get({
15            key: ZIP_TO_CITY_IDX_JSON,
16            loader: lib.zipCodeDatabaseLoader
17        });
18        return JSON.parse(zipCacheJson);
19    }
20
21    function findCityByZipCode(options) {
22        return getZipCodeToCityLookupObj()[String(options.zip)];
23    }
24
25    function onRequest(context) {
26        var start = new Date();
27        if (context.request.parameters.purgeZipCache === 'true') {
28            var zipCache = cache.getCache({
29                name: ZIP_CODES_CACHE_NAME
30            });
31            zipCache.remove({
32                key: ZIP_TO_CITY_IDX_JSON
33            });
34        }
35        var cityName = findCityByZipCode({
36            zip: context.request.parameters.zipcode
37        });
38        context.response.writeLine(cityName || 'Unknown :(');
39        if (context.request.parameters.auditPerf === 'true') {
40            context.response.writeLine('Time Elapsed: ' + (new Date().getTime() - start.getTime()) + ' ms');
41        }
}

```

```

42     }
43
44     return {
45       onRequest: onRequest
46     };
47 });

```

The following custom module returns the loader function used in the preceding script sample. The loader function shows how to use a CSV file to retrieve a value that was missing from a cache. This custom module does not need to include logic for placing the retrieved value into the cache — whenever a value is returned through the options.loader parameter, the value is automatically placed into the cache. For this reason, a loader function can serve as the sole method of populating a cache with values.

```

1 /**
2 * zipToCityIndexCacheLoader.js
3 * @NApiVersion 2.x
4 * @NModuleScope Public
5 */
6 //This is a loader function that uses a CSV file to retrieve a value that was missing from a cache.
7 define(['N/file', 'N/cache'], function(file, cache) {
8   const ZIP_CODES_CSV_PATH = '/SuiteScripts/Resources/free-zipcode-CA-database-primary.csv';
9
10  function trimOuterQuotes(str) {
11    return (str || '').replace(/^\+/ , '').replace(/\+$/ , '');
12  }
13
14  function zipCodeDatabaseLoader(context) {
15    log.audit('Loading Zip Codes for ZIP_CODES_CACHE');
16    var zipCodesCsvText = file.load({
17      id: ZIP_CODES_CSV_PATH
18    }).getContents();
19    var zipToCityIndex = {};
20    var csvLines = zipCodesCsvText.split('\n');
21    util.each(csvLines.slice(1), function(el) {
22      var cells = el.split(',');
23      var key = trimOuterQuotes(cells[0]);
24      var value = trimOuterQuotes(cells[2]);
25      if (parseInt(key, 10))
26        zipToCityIndex[String(key)] = value;
27    });
28    return zipToCityIndex;
29  }
30
31  return {
32    zipCodeDatabaseLoader: zipCodeDatabaseLoader
33  };
34 });

```

cache.Cache



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A segment of memory that can be used to temporarily store data (on a short term basis) needed by one script, by all scripts in a bundle, or by all scripts in the NetSuite account. This object is returned by cache.getCache(options) .
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/cache Module
Methods and Properties	Cache Object Members

Since	2016.2
-------	--------

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/cache Module Script Sample](#).

```

1 //Add additional code
2 ...
3 //myCache in the following statement will be a cache.Cache object as returned from cache.getCache
4 var myCache = cache.getCache({
5   name: 'temporaryCache',
6   scope: cache.Scope.PRIVATE
7 });
8 ...
9 //Add additional code

```

Cache.get(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves a string value from the cache. The value retrieved is identified by a key that you pass by using the options.key parameter. If a requested value is not present in the cache, the system calls the function identified by the options.loader parameter. This user-defined function should provide logic for retrieving a value that is not in the cache. For an example, see N/cache Module Script Sample .
Returns	String or null
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	1 unit if the value is present in the cache; 2 units if the loader function is used
Module	N/cache Module
Parent Object	cache.Cache
Sibling Object Members	Cache Object Members
Since	2016.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.key	string	required	A string that identifies the value to be retrieved from the cache. This value cannot be null.
options.loader	function	optional, but strongly recommended	A user-defined function that returns the requested value if it is not already present in the cache. Additionally, when the loader retrieves a value, the system automatically places that value

Parameter	Type	Required / Optional	Description
			<p>in the cache. For this reason, NetSuite recommends using the loader function as the primary means of populating the cache. For an example, see N/cache Module Script Sample.</p> <p>Note also that if the value returned by the loader is not a string, the system uses JSON.stringify() to convert the value before it is placed in the cache and returned. The maximum size of a value that can be placed in the cache is 500KB.</p> <p>When no loader is specified and a value is missing from the cache, the system returns null.</p>
options.ttl	number	optional	<p>The maximum duration, in seconds, that a value retrieved by the loader can remain in the cache. Note that the value may be removed from the cache before the ttl limit is reached.</p> <p>The minimum value is 300 (five minutes) and there is no maximum. The default ttl value is no limit.</p> <div style="border: 1px solid #f0e68c; padding: 10px; margin-top: 10px;"> ⚠️ Important: A cached value is not guaranteed to stay in the cache for the full duration of the ttl value. The ttl value represents the maximum time that the cached value may be stored. </div>

Syntax

⚠️ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/cache Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myCache = cache.getCache({
4   name: 'temporaryCache',
5   scope: cache.Scope.PRIVATE
6 });
7
8 var myValue = myCache.get({
9   key: 'keyText',
10  loader,
11  ttl: 18000
12 });
13 ...
14 //Add additional code

```

Cache.put(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Puts a value into the cache.
	<p>i Note: You can also put a value in a cache by using the Cache.get(options) method and the options.loader parameter. In general, using the get method is recommended and may result in a more efficient design. For an example, see N/cache Module Script Sample</p>
Returns	void

Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	1 unit
Module	N/cache Module
Parent Object	cache.Cache
Sibling Object Members	Cache Object Members
Since	2016.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.key	string	required	The identifier of the value that is being cached. The maximum size of the cache key is 4 kilobytes.
options.value	string	required	The value to place in the cache. If the value submitted is not a string, the system uses <code>JSON.stringify()</code> to convert the value before it is placed in the cache. The maximum size of the value is 500KB.
options.ttl	number	optional	<p>The maximum duration, in seconds, that the value may remain in the cache. Note that the value may be removed before the ttl limit is reached.</p> <p>The minimum value is 300 (five minutes) and there is no maximum. The default ttl value is no limit.</p> <p> Important: A cached value is not guaranteed to stay in the cache for the full duration of the ttl value. The ttl value represents the maximum time that the cached value may be stored. Cached data is not persistent, and it is recommended that you use the Cache.get(options) method and options.loader parameter to set and retrieve data.</p>

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/cache Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myCache = cache.getCache({
4   name: 'temporaryCache',
5   scope: cache.Scope.PRIVATE
6 });
7 myCache.put({
8   key: 'keyText',
9   value: 'valueText',
10  ttl: 300
11 });

```

```

12 | ...
13 //Add additional code

```

Cache.remove(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes a value from the cache.
Returns	void
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	1 unit
Module	N/cache Module
Parent Object	cache.Cache
Sibling Object Members	Cache Object Members
Since	2016.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.key	string	required	The identifier of the value that is being removed.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/cache Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myCache = cache.getCache({
4   name: 'temporaryCache',
5   scope: cache.Scope.PRIVATE
6 });
7 myCache.remove({
8   key: 'keyText'
9 });
10 ...
11 //Add additional code

```

Cache.name

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A label that identifies a cache.
-----------------------------	----------------------------------

Type	string
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/cache Module
Parent Object	cache.Cache
Sibling Object Members	Cache Object Members
Since	2016.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/cache Module Script Sample](#)

```

1 //Add additional code
2 ...
3 var myCache = cache.getCache({
4   name: 'temporaryCache', //Cache.name
5   scope: cache.Scope.PRIVATE
6 });
7 ...
8 //Add additional code

```

Cache.scope

ℹ Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A label that describes the availability of the cache to other scripts.
Type	cache.Scope
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/cache Module
Parent Object	cache.Cache
Sibling Object Members	Cache Object Members
Since	2016.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/cache Module Script Sample](#)

```

1 //Add additional code
2 ...

```

```

3 var myCache = cache.getCache({
4   name: 'temporaryCache',
5   scope: cache.Scope.PRIVATE //Cache.scope
6 });
7 ...
8 //Add additional code

```

cache.getCache(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Checks for a cache object with the specified name. If the cache exists, this method returns the cache object. If the cache does not exist, the system creates it.
Returns	<code>cache.Cache</code>
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	n/a
Module	N/cache Module
Sibling Module Members	N/cache Module Members
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.name</code>	string	required	A label identifies the cache to be retrieved or to be created. The maximum size of the cache name is 1 kilobyte.
<code>options.scope</code>	string	optional, if you do not set a value, the default value PRIVATE is used	This value is set with the <code>cache.Scope</code> enum. It determines the availability of the cache. A cache can be made available to the current script only, to all scripts in the current bundle, or to all scripts in your NetSuite account.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/cache Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myCache = cache.getCache({
4   name: 'temporaryCache',
5   scope: cache.Scope.PRIVATE
6 });

```

```

7 ...
8 //Add additional code

```

cache.Scope



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Enumeration that holds string values that describe the availability of the cache. This enum is used to set the value of the Cache.scope property.
Type	enum
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/cache Module
Sibling Module Members	N/cache Module Members
Since	2016.2

Values

Value	Description
PRIVATE	The cache is available only to the current script. This value is the default.
PROTECTED	The cache is available only to some scripts, as follows: <ul style="list-style-type: none"> ■ If the script is part of a bundle, the cache is available to all scripts in the same bundle. ■ If the script is not in a bundle, the cache is available to all scripts not in any bundle.
PUBLIC	The cache is available to any script in the NetSuite account.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/cache Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myCache = cache.getCache({
4   name: 'temporaryCache',
5   scope: cache.Scope.PRIVATE
6 });
7 ...
8 //Add additional code

```

N/certificateControl Module



Note: The content in this help topic pertains to SuiteScript 2.0.

The N/certificateControl module enables scripting access to the Digital Certificates list found in the UI at Setup > Company > Certificates. You can use this module to find, create, update, read and delete certificates records. For more information, see the help topics [Digital Signing](#) and [Uploading Digital Certificates](#).

In order to access this module, you must use the Execute As Role field on the script deployment record. Select either the Administrator role or a custom role with the Certificate Access permission. For more information, see the help topic [Access to Digital Certificates](#).



Important: The certificate record holds information for a digital certificate, but it is not a standard NetSuite record and cannot be accessed with the N/record.

N/certificateControl Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	certificateControl.Certificate	object	server scripts	Encapsulates a digital certificate record.
Method	certificateControl.findCertificates(options)	object	server scripts	Returns metadata about the certificate(s).
	certificateControl.findUsages(options)	object[]	server scripts	Returns an audit trail of how a certificate has been used. Includes operations performed with time stamps.
	certificateControl.createCertificate(options)	certificateControl.Certificate	server scripts	Creates a certificate record using a file from the File Cabinet. After saving with Certificate.save() , the certificate is accessible on the Certificates .
	certificateControl.deleteCertificate(options)	string	server scripts	Deletes a certificate record that has been uploaded to the Certificates list in the UI or created using certificateControl.createCertificate(options) and saved with Certificate.save() .
	certificateControl.loadCertificate(options)	certificateControl.Certificate	server scripts	Loads a certificate record that has been uploaded to the Certificates list in the UI or created using certificateControl.createCertificate(options) .
Enum	certificateControl.Type	enum	server scripts	Enum for certificate types. PFX, PEM, and P12 are supported types.
	certificateControl.Operation	enum	server scripts	Enum for searching the audit trail of certificates with certificateControl.findUsages(options) .
	certificateControl.Operator	enum	server scripts	Enum for searching for certificate records with certificateControl.findCertificates(options) .

Certificate Object Members

The following members are called on the `certificateControl.Certificate` object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	<code>Certificate.save()</code>	object containing the script ID of the new certificate record	server scripts	Saves a certificate record.
Property	<code>Certificate.description</code>	string	server scripts	Describes the certificate record.
	<code>Certificate.file</code>	File Object Members object	server scripts	Includes the properties of the file uploaded to create the certificate.
	<code>Certificate.name</code>	string	server scripts	The name of the certificate record.
	<code>Certificate.monthReminder</code>	boolean true false	server scripts	Indicates the setting of the Month box for Expiration Reminders on the certificate record.
	<code>Certificate.notifications</code>	number[]	server scripts	The internal IDs of the employees selected in the Copy Employees field on the certificate record.
	<code>Certificate.password</code>	string (write-only)	server scripts	The password for the digital certificate. You can create a GUID using Form.addSecretKeyField (options) .
	<code>Certificate.restrictions</code>	number[]	server scripts	The internal IDs of the employees selected in the Restrict to Employees field of the certificate record.
	<code>Certificate.scriptId</code>	string	server scripts	The ID of the certificate record.
	<code>Certificate.subsidiaries</code>	number[]	server scripts	The internal IDs of the subsidiaries associated with the certificate record.
	<code>Certificate.threeMonthsReminder</code>	boolean true false	server scripts	Indicates the setting of the 3 Months box for Expiration Reminders on the certificate record.
	<code>Certificate.weekReminder</code>	boolean true false	server scripts	Indicates the setting of the Week box for Expiration Reminders on the certificate record.

N/certificateControl Module Script Samples

Example 1

i Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to filter the Digital Certificates list by subsidiary and by file type.

```

1 /**
2  * @NApiVersion 2.x
3 */
4 require(['N/certificateControl'],
5         function(certificateControl){
6             var all = certificateControl.findCertificates();
7             var specificType = certificateControl.findCertificates({
8                 type: 'PFX'
9             });
10            var specificSub = certificateControl.findCertificates({
11                subsidiary: 93
12            });
13            var specificTypeAndSub = certificateControl.findCertificates({
14                type: 'PFX',
15                subsidiary: 93
16            });

```

Example 2

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to find the audit trail of POST operations for the certificate record with ID 'custcertificate_china'.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/certificateControl'], function(cc){
6     var usages = cc.findUsages({
7         id: 'custcertificate_china',
8         operation: cc.Operation.POST
9     });
10})

```

Example 3

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to create a file object by loading a file from the File Cabinet. It then creates the options needed for the `certificateControl.createCertificate(options)` method and creates and saves the certificate record. The certificate record is then loaded again, edited to change the file, and saved again.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/certificateControl', 'N/file'], function(cc, file){
6     var fileObj = file.load({
7         id: 'SuiteScripts/dsa.p12'
8     });
9     var options = {
10         file : fileObj,

```

```

11     password : '022b490ad4334c7e86a8304f937ec68f',
12     name : 'testCert',
13     description : 'testDescription',
14     scriptId : '_testid',
15     subsidiaries : [1,3],
16     weekReminder : false,
17     monthReminder : true,
18     threeMonthsReminder : false
19   };
20   var newCertificate = cc.createCertificate(options);
21   newCertificate.save();
22
23   var loadedCertificate = cc.loadCertificate({
24     scriptId : 'custcertificate_testid'
25   });
26   fileObj = file.load({
27     id: 'SuiteScripts/ecdsa.p12'
28   });
29   loadedCertificate.file = fileObj;
30   loadedCertificate.password = '022b490ad4334c7e86a8304f937ec68f';
31   loadedCertificate.save();
32 })

```

Example 4

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to find an existing certificate record and use it in an operation.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/certificateControl', 'N/https/clientCertificate'],function(cc, cert){
6   var yodlee = cc.findCertificates({
7     name: 'Yodlee',
8     description: 'Yodlee certificate'
9   });
10  cert.post({certId:yodlee[0].id,url:<url>, body:<body>, headers:<headers>
11 });
12 })

```

Example 5

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to generate a signature of a plaintext string and then verifies the signature using the same certificate.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/certificateControl', 'N/crypto/certificate'], function(cc, certificate){
6   var signer = certificate.createSigner({
7     certId:'custcertificate_cert_1',
8     algorithm: 'SHA256'

```

```

9     });
10    var result = signer.sign();
11    var verifier = certificate.createVerifier({
12      certId: 'custcertificate_cert_1',
13      algorithm: 'SHA256'
14    });
15    verifier.update('test');
16    verifier.verify(result);
17    var res = cc.findUsages();
18    ;
19 })
)

```

The `res` variable returns an array of information about the usage of the digital certificate, including the date of the action, the type of operation, such as sign, and the internal ID of the person who performed the action.

Example 6

i Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample establishes a SFTP connection using an SSH key that has already been uploaded to NetSuite. It then creates, updates, loads, and deletes a certificate record to show the full CRUD operation. Replace the server URL with your correct URL.

For the SFTP connection, the public key corresponding to the private key in the certificate must be stored in the `.ssh/authorized_keys` file on the server.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/file', 'N/sftp', 'N/certificateControl'], function(file, sftp, certificateControl) {
6   var certPath = 'yyy/certificates'
7   var certName = 'apiclient_cert.p12';
8
9 // Establish SFTP connection
10  log.debug('Establishing secured SFTP connection...');
11  var connection = sftp.createConnection({
12    username: 'sftpuser',
13    keyId: 'custkeysftp_nft_demo_key',
14    url: 'my.sftp.server.com',
15    port: 22,
16    directory: 'inbound',
17    hostKey: 'AAAAAB3NzaC1yc2EAAAQABAAQAC4gYD1K41E9QnuYEgRRQChjrzAM1+bTT95e71Xv0oQ60ywVQEiedhRqSMbPiCPPB4pjBdOmPIQC
18 Ckug+3XwAq6uNj3UM1zoGGmp86tyEJt6qGB0SsrQJzHtB3EG38BSrB00WEz0WeJ8E8Y0DT3oAj1Nrfls3JbG0RF+0uwJD1l1sFkYS3kWCv27NhBnaytGe7iLBgrJd
19 NVlitQkxZfK0NsAYCaJW0jQLtz+GFFN5zTbKKNsDa6s/YW7oAMMOI3Q5GQAqdXtKY728WvxYTjr2FsYS/KM6nbq/csTvZHwLE0z2TQtB2H0IIofvEP/QvXwmqEnCeVPCn
gRwdHWQf'
20  });
21  log.debug('Connection established!');
22 // -----
23 // Create new certificate
24  log.debug('Creating certificate...');
25  var certscriptId = '_' + (Math.random().toString(36).substring(2, 10));
26  var cert = certificateControl.createCertificate();
27  cert.name = 'Test Certificate China API';
28  cert.description = 'Test Certificate China created via API';
29 // custcertificate prefix will be added automatically
30  cert.scriptId = certscriptId;
31  log.debug('Downloading certificate file from SFTP...');
32  cert.file = connection.download({
33    directory: certPath,
34    filename: certName
35  });

```

```

35     log.debug('Successfully downloaded!');
36     //guid corresponding to the certificate's password;
37     var pwd = '022b490ad4334c7e86a8304f937ec68f',
38     cert.password = pwd;
39     cert.save();
40 /**
41 **/certScriptId = 'custcertificate' + certScriptId;
42 log.debug("Certificate " + cert.name + " successfully created with id " + certScriptId + "!");
43 //-----
44 // Rename certificate
45 log.debug('Renaming certificate...');
46 cert = certificateControl.loadCertificate({scriptId: certScriptId});
47 cert.name = 'Test Certificate China API TEMP';
48 cert.save();
49 //Verify new certificate name
50 /**
51 **/cert = certificateControl.loadCertificate({scriptId: certScriptId});
52 log.debug('Certificate successfully renamed to "' + cert.name + "'");
53 // -----
54 // Delete certificate
55 log.debug('Deleting certificate "' + cert.name + '"...');
56 certificateControl.deleteCertificate(certScriptId);
57 log.debug('Certificate deleted!');
58 // -----
59 // Load the deleted certificate
60 log.debug('Attempting to load the deleted certificate to invoke an error...');
61 try {
62     cert = certificateControl.loadCertificate({scriptId: certScriptId});
63 } catch (e) {
64     log.error(e.message);
65 }
66 })

```

certificateControl.Certificate



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The certificate record, including file name and preferences.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/certificateControl Module
Methods and Properties	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var loadedCertificate = cc.loadCertificate({
4     scriptId : 'custcertificate_testid'
5 });
6 fileObj = file.load({
7     id: 'SuiteScripts/ecdsa.p12'
8 });
9 loadedCertificate.file = fileObj;

```

```

10 loadedCertificate.password = '022b490ad4334c7e86a8304f937ec68f',
11 loadedCertificate.save();
12 cc.deleteCertificate({
13   scriptId : 'custcertificate_testid'
14 });
15 ...
16 // Add additional code

```

Certificate.save()

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Saves a certificate record object.
Returns	certificateControl.Certificate object
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 fileObj = file.load({
4   id: 'SuiteScripts/ecdsa.p12'
5 });
6 loadedCertificate.file = fileObj;
7 loadedCertificate.password = '022b490ad4334c7e86a8304f937ec68f',
8 loadedCertificate.save();
9 ...
10 // Add additional code

```

Certificate.description

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A description of the certificate record.
Type	string
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate

Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3
4 //load the certificate record object
5 var loadedCertificate = cc.loadCertificate({
6  scriptId : 'custcertificate_testid'
7 });
8 //update the description for the certificate record
9 loadedCertificate.description = 'Test Certificate Description'
10 //save the updated certificate record
11 loadedCertificate.save();
12 ...
13 // Add additional code

```

Certificate.file



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The File Object Members object of the certificate uploaded to the certificate record.
Type	File Object Members object
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3
4 //load the certificate record object
5 var loadedCertificate = cc.loadCertificate({
6  scriptId : 'custcertificate_testid'
7 });
8 //load the file from the File Cabinet
9 fileObj = file.load({
10   id: 'SuiteScripts/ecdsa.p12'
11 });
12 //upload the file to the certificate record
13 loadedCertificate.file = fileObj;
14 //save the certificate record
15 loadedCertificate.save();

```

```

15 | ...
16 | //Add additional code

```

Certificate.name



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of the certificate record.
Type	string
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //load the certificate record object
4 var loadedCertificate = cc.loadCertificate({
5  scriptId : 'custcertificate_testid'
6 });
7 //update the name of the certificate record
8 loadedCertificate.name = 'Brazil Certificate';
9 //save the certificate record object
10 loadedCertificate.save();
11 ...
12 // Add additional code

```

Certificate.monthReminder



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates the setting of the Month box for Expiration Reminders on the certificate record. This property is set to true if the Month box is checked and email reminders are sent to account administrators one month before the certificate expires. If the Copy Employees box is also checked, selected employees are copied on the reminder emails.
Type	boolean true false
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members

Since	2019.2
-------	--------

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //load the certificate record object
4 var loadedCertificate = cc.loadCertificate({
5  scriptId : 'custcertificate_testid'
6 });
7 //update the Expiration Reminder for Month to checked
8 loadedCertificate.monthReminder = true;
9 //save the certificate record object
10 loadedCertificate.save();
11 ...
12 // Add additional code

```

Certificate.notifications

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal IDs of the employees copied on expiration notification email. The values for this property are found in the Copy Employees field of the Audience tab on the certificate record. When you create or edit a certificate object with values for this property, you also check the Copy Employees box for the certificate record.
Type	number []
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //create a variable to hold the properties of the certificate object
4 var options = {
5   name : 'testCertp12',
6   description : 'testDescription',
7   scriptId : '_testidp12',
8   //Include the internal IDs for employees you want copied on expiration reminder email
9   notifications: [168,259]

```

```

10  };
11 //create the certificate record with the options variable
12 var newCertificate = cc.createCertificate(options);
13 //save the certificate object
14 newCertificate.save();
15 ...
16 // Add additional code

```

Certificate.password



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The password for the digital certificate. If the certificate file is password-protected, you can store the password with the certificate record. If the certificate is not password-protected, enter an empty string.
Type	string (write-only)
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //create a variable to hold the properties of the certificate object
4 var options = {
5   name : 'testCertp12',
6   description : 'testDescription',
7   //for password, enter the guid associated with your digital certificate or an empty string
8   password: 'yourCertPassword',
9  scriptId : '_testidp12',
10 };
11 //create the certificate record with the options variable
12 var newCertificate = cc.createCertificate(options);
13 //save the certificate object
14 newCertificate.save();
15 ...
16 // Add additional code

```

Certificate.restrictions



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal IDs of the employees selected in the Restrict to Employees field of the certificate record. If you set this property with an employee internal ID, you check the Restrict to Employees box and select that employee.
-----------------------------	---

	Employees selected must also have either the Certificate Management or Certificate Access role permission in order to access the certificate. When the Restrict to Employees box is checked, only Administrators and the employees selected can access the certificate.
Type	number[]
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //load the certificate record object
4 var loadedCertificate = cc.loadCertificate({
5  scriptId : 'custcertificate_testid'
6 });
7 //check the Restrict to Employees box and select employees with internal IDs of 189 and 250
8 loadedCertificate.restrictions = [189,250];
9 loadedCertificate.save();
10 ...
11 // Add additional code

```

Certificate.scriptId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the certificate record. The script ID for certificate records begins with "custcertificate."
Type	string
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //load the certificate record object

```

```

4 var loadedCertificate = cc.loadCertificate({
5   scriptId : 'custcertificate_testid'
6 });
7 //update the text for script ID
8 loadedCertificate.scriptId = '_ChinaCert';
9 loadedCertificate.save();
10 ...
11 //Add additional code

```

Certificate.subsidiaries



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal IDs of the subsidiaries associated with the certificate record. Subsidiary selections associate a certificate to one or more subsidiaries but do not affect access.
Type	number[]
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //load the certificate record object
4 var loadedCertificate = cc.loadCertificate({
5   scriptId : 'custcertificate_testid'
6 });
7 //set the subsidiaries to those with the internal IDs of 3 and 5
8 loadedCertificate.subsidiaries = [3,5];
9 //save the certificate record object
10 loadedCertificate.save();
11 ...
12 // Add additional code

```

Certificate.threeMonthsReminder



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates the setting of the 3 Months box for Expiration Reminders on the certificate record. This property is set to true if the 3 Months box is checked. When set to true , email reminders are sent to account administrators three months before the certificate expires. If the Copy Employees box is also checked, selected employees are copied on the reminder emails.
Type	boolean true false

Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //load the certificate record object
4 var loadedCertificate = cc.loadCertificate({
5  scriptId : 'custcertificate_testid'
6 });
7 //update the Expiration Reminder for 3 Months to checked
8 loadedCertificate.threeMonthsReminder = true;
9 //save the certificate record object
10 loadedCertificate.save();
11 ...
12 // Add additional code

```

Certificate.weekReminder



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates the setting of the Week box for Expiration Reminders on the certificate record. This property is set to true if the Week box is checked. When set to true , email reminders are sent to account administrators one week before the certificate expires. If the Copy Employees box is also checked, selected employees are copied on the reminder emails.
Type	boolean true false
Module	N/certificateControl Module
Parent Object	certificateControl.Certificate
Sibling Object Members	Certificate Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...

```

```

3 //load the certificate record object
4 var loadedCertificate = cc.loadCertificate({
5  scriptId : 'custcertificate_testid'
6 });
7 //update the Expiration Reminder for Week to checked
8 loadedCertificate.weekReminder = true;
9 //save the certificate record object
10 loadedCertificate.save();
11 ...
12 // Add additional code

```

certificateControl.createCertificate(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a certificate record on the Certificates page using a file from the File Cabinet.  Note: Your role must have Create, Edit, or Full access to the Certificate Access permission to create certificates via SuiteScript.
Returns	certificateControl.Certificate
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/certificateControl Module
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.file	file	required	A File Object Members object. The file must already be uploaded to the File Cabinet.	2019.2
options.password	string	optional	If applicable, the password associated with your digital certificate.	2019.2
options.scriptId	string	optional	The desired script ID of the certificate record. The script ID is automatically prefixed with 'custcertificate_'	2019.2
options.description	string	optional	The description of the certificate record.	2019.2
options.subsidiaries	number[] or string[]	optional	The internal ID of subsidiaries associated with the certificate in either number or string format.	2019.2
options.restrictions	number[] or string[]	optional	The internal ID of employees selected in the Restricted to Employees field for a	2019.2

Parameter	Type	Required / Optional	Description	Since
			certificate. You can enter the internal ID in either number or string format.	
options.notifications	number[] or string[]	optional	The internal ID of employees selected in the Copy Employees field on the certificate record. You can enter the internal ID in either number or string format.	2019.2
options.name	string	required	The name of the certificate record.	2019.2
options.weekReminder	boolean true false	optional	The setting for the Expiration Reminder : Week checkbox.	2019.2
options.monthReminder	boolean true false	optional	The setting for the Expiration Reminder : Month checkbox.	2019.2
options.threeMonthsReminder	boolean true false	optional	The setting for the Expiration Reminder : 3 Months checkbox.	2019.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 'SuiteScripts/dsa.p12'
5 });
6 var options = {
7   file : fileObj,
8   password : '022b490ad4334c7e86a8304f937ec68f',
9   name : 'testCert',
10  description : 'testDescription',
11 scriptId : '_testid',
12  subsidiaries : [1,3],
13  weekReminder : false,
14  monthReminder : true,
15  threeMonthsReminder : false
16 };
17 var newCertificate = cc.createCertificate(options);
18 newCertificate.save();
19 ...
20 //Add additional code

```

certificateControl.findCertificates(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an array of certificates available. You can use the parameters as filters for this search. If you do not use any parameters, all certificate records are returned.
Returns	Metadata about the certificate(s)
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .

Governance	10 units
Module	N/certificateControl Module
Since	2019.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.subsidiary	number	optional	The internal ID of the subsidiary.	2019.1
options.type	string	optional	The certificate file type.	2019.1
options.restriction	number	optional	The internal ID of an employee selected in the Restrict to Employees field.	2019.2
options.notification	number	optional	The internal ID of an employee selected in the Copy Employees field.	2019.2
options.name	string	optional	The certificate name. You can use this filter with the certificateControl.Operator enum.	2019.2
options.description	string	optional	The certificate description. You can use this filter with the certificateControl.Operator enum.	2019.2

Syntax

The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 | require(['N/certificateControl'],function(cc){
2 |   var yodlee = cc.findCertificates({
3 |     name: 'Yodlee',
4 |     description: 'Yodlee certificate'
5 |   });
6 | })

```

certificateControl.findUsages(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an audit trail of how a certificate has been used. Includes operations performed with time stamps.
Returns	An array of operations performed.
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units

Module	N/certificateControl Module
Since	2019.2

Parameters

Note:	The options parameter is a JavaScript object.
--------------	---

Parameter	Type	Required / Optional	Description	Since
options.from	date	optional	The start date for your audit trail search.	2019.2
options.to	date	optional	The end date for your audit trail search.	2019.2
options.id	string	optional	The script ID of the certificate record.	2019.2
options.operation	string	optional	The certificateControl.Operation performed with the digital certificate.	2019.2
options.script	number	optional	The script ID of a script record that used a certificate record.	2019.2
options.deploy	number	optional	The script ID of a script deployment that used a certificate record.	2019.2
options.entity	number	optional	The internal ID of the employee who performed the operation.	2019.2

Error	Thrown If
SSS_INVALID_TYPE_ARG	A parameter provided is the wrong type.
TOO_MANY_RESULTS	There are more than 1000 results.

Syntax

The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var usages = cc.findUsages({
4   id: 'custcertificate_china',
5   operation: cc.Operation.POST
6 });
7 ...
8 //Add additional code

```

certificateControl.deleteCertificate(options)

Note:	The content in this help topic pertains to SuiteScript 2.0.
--------------	---

Method Description	Deletes a certificate record that has been uploaded to the Certificates list in the UI or created using certificateControl.createCertificate(options) .
---------------------------	---

	<p>Note: Your role must have either Edit or Full access to the Certificate Access permission to delete certificate records via SuiteScript. History of the certificate is not deleted.</p>
Returns	The script ID of the deleted certificate.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/certificateControl Module
Since	2019.2

Parameters

Note: The options parameter is a JavaScript object.								
<table border="1"> <thead> <tr> <th>Parameter</th> <th>Type</th> <th>Required / Optional</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>options.scriptId</td> <td>string</td> <td>required</td> <td>The script ID or internal ID for the certificate you want to delete. You can view the ID of a certificate from the Digital Certificates list at Setup > Company > Certificates.</td> </tr> </tbody> </table>	Parameter	Type	Required / Optional	Description	options.scriptId	string	required	The script ID or internal ID for the certificate you want to delete. You can view the ID of a certificate from the Digital Certificates list at Setup > Company > Certificates.
Parameter	Type	Required / Optional	Description					
options.scriptId	string	required	The script ID or internal ID for the certificate you want to delete. You can view the ID of a certificate from the Digital Certificates list at Setup > Company > Certificates.					

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 define(['N/certificateControl'], function(cc){
4     var usages = cc.deleteCertificate({
5        scriptId: 'custcertificate_china'
6     });
7 }
8 ...
9 // Add additional code

```

certificateControl.loadCertificate(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
Method Description
Loads a certificate record that has been uploaded to the Certificates list in the UI or created using certificateControl.createCertificate(options) .
Returns
certificateControl.Certificate
Supported Script Types
Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/certificateControl Module
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.scriptId	string	required	The script ID or internal ID for the certificate you want to load. You can view the ID of a certificate from the Digital Certificates list at Setup > Company > Certificates.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //load the certificate record object
4 var loadedCertificate = cc.loadCertificate({
5  scriptId : 'custcertificate_testid'
6 });
7 //load a digital certificate from the File Cabinet
8 fileObj = file.load({
9   id: 'SuiteScripts/ecdsa.p12'
10 });
11 //upload the file to the certificate record
12 loadedCertificate.file = fileObj;
13 //update the password to match the guid password for the certificate
14 loadedCertificate.password = 'certPass';
15 //save the certificate
16 loadedCertificate.save();
17 ...
18 //Add additional code

```

certificateControl.Operation

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

 **Note:** JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Enumeration that holds the values for the operation parameter of .
-------------------------	--

Module	N/certificateControl Module
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Since	2019.2

Values

- CONNECT
- DELETE
- FIND
- GET
- HEAD
- POST
- PUT
- SIGN_STRING
- SIGN_XML
- VERIFY_STRING
- VERIFY_XML

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var usages = cc.findUsages({
4   id: 'custcertificate_china',
5   operation: cc.Operation.POST
6 });
7 ...
8 //Add additional code

```

certificateControl.Operator



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Search operators to use with the name and description parameters of the certificateControl.findCertificates(options) .
Module	N/certificateControl Module

Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Since	2019.2

Values

- CONTAINS
- ENDS_WITH
- EQUALS
- STARTS_WITH

certificateControl.Type

i Note: The content in this help topic pertains to SuiteScript 2.0.

i Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	The certificate file type. PFX, PEM, and P12 are supported.
Module	N/certificateControl Module
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Since	2019.1

Values

- PFX
- P12
- PEM

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/certificateControl Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var specificType = certificateControl.findCertificates({
4     type: certificateControl.Type.PFX
5 });
6 ...
7 // Add additional code

```

Ncommerce Modules

ⓘ Applies to: Commerce Web Stores

ⓘ Note: The content in this help topic pertains to SuiteScript 2.0.

Modules in the Ncommerce namespace have been designed for use with SSP applications written in SuiteScript 2.x. Their primary objective is to provide web developers a homogeneous development experience when working with Commerce and SuiteScript APIs. Note that Ncommerce itself is not a module.

Developers can use modules in the Ncommerce namespace to access different assets in the web store context, such as items and shopping cart. The modules within the Ncommerce namespace are supported by the latest version of SuiteCommerce and by SuiteCommerce Advanced 2019.2 onwards.

Before you can load Ncommerce modules, you must have an active shopping session.

The modules available within the Ncommerce namespace are:

Module	Description
Ncommerce/recordView Module	Load the Ncommerce/recordView module when you want to provide fast, cached, and public access to the item fields and website settings.

ⓘ Note: Commerce modules are not yet available for all web store interactions. To fully customize your web store on the SuiteCloud platform, you can use Commerce APIs with SSP applications written in SuiteScript 1.0. However, Commerce APIs are not available for use with SSP applications written in SuiteScript 2.x. For more information about the Commerce APIs, see the help topic [Commerce API](#).

Ncommerce/recordView Module

ⓘ Applies to: Commerce Web Stores

ⓘ Note: The content in this help topic pertains to SuiteScript 2.0.

Load the Ncommerce/recordView module when you want to provide fast, cached, and public access to the item fields and website settings.

- [Ncommerce/recordView Module Members](#)
- [Ncommerce/recordView Script Sample](#)

Ncommerce/recordView Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	recordView.viewItems (options)	Object array	Client and server-side scripts	Retrieves one or more Items with requested

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
		Returns one or more Items with requested items fields as an object with field:value pairs. See the Returns section.		items fields from an Item Record.
	recordView.viewWebsite(options)	Object Returns website and website fields as an object with field:value pairs. See the Returns section.	Client and server-side scripts	Retrieves the website details with requested website fields.

Ncommerce/recordView Script Sample

The following example retrieves some details of the website and some item data for the specified items.

Note: This sample script uses the define function, which is used in your entry point script (the script you attach to a script record). Keep in mind that you must use the require function instead of define if you want to copy it into the debugger and test it.

For help with writing scripts in SuiteScript 2.x, see [SuiteScript 2.0 Hello World](#) and [SuiteScript 2.0 Entry Point Script Creation and Deployment](#).

```

1  /**
2  * @NApiVersion 2.x
3 */
4 define(['Ncommerce/recordView'],
5     function (recordView) {
6         function service(context) {
7             var result = {};
8
9             try {
10                 result.website= recordView.viewWebsite({
11                     id: 2,
12                     fields: ["internalid","shiptocountries"]
13                 });
14             }
15             catch (e) {
16                 result.websiteError = e.name + " : " + e.message;
17             }
18
19             var options = {
20                 "ids": [382,388],
21                 "fields": ["displayname"]
22             };
23             try {
24                 result.items= recordView.viewItems(options);
25             }
26             catch (e) {
27                 result.itemsError = e.name + " : " + e.message;
28             }
29             return context.response.write(
30                 JSON.stringify(result)
31             );
32         }
33
34         return {
35             service: service
36         };
}

```

```

37     }
38 );

```

recordView.viewItems(options)

i Applies to: Commerce Web Stores

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves one or more items with requested item fields from an Item Record.
Returns	See the Returns section.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	Ncommerce/recordView Module
Sibling Module Members	Ncommerce/recordView Module Members
Since	2019.1

Parameters

i Note: The options parameter is a JavaScript object.

Property	Type	Required / Optional	Description
options.ids	number[]	required	IDs of the item you want to view
options.fields	string string[]	required	Item fields you want to retrieve for the items. For more information, see Supported Fields .
options.fieldOptions	Array of name, value pairs. Type depends upon parameter: includeVat - string boolean[]	optional	Options that affect related fields. Supported field options: includeVat - this affects onlinecustomerprice_detail field. Default value is false.

Supported Fields

The following fields can be requested in the options.fields parameter:

Record Fields			
amortizationperiod	enforceminqtyinternally	mpn	searchkeywords
amortizationtemplate	excludedefromsitemap	nextagcategory	seasonaldemand
assetaccount	expenseaccount	nopricemessage	shippingcost
autoleadtime	externalid	outofstockmessage	shoppingdotcomcategory

autopreferredstocklevel	featureddescription	overallquantitypricingtype	shopzillacategoryid
autoreorderpoint	gainlossaccount	pagetitle	showdefaultdonationamount
availabletopartners	generateaccruals	parent	stockdescription
averagecost	handlingcost	preferredstocklocation	stockunit
billexchratevarianceacct	incomeaccount	preferredstocklevel	storedescription
billingschedule	internalid	preferredstockleveldays	storedetaileddescription
billpricevarianceacct	isdonationitem	pricinggroup	storedisplayimage
billqtyvarianceacct	isdropshipitem	printitems	storedisplayname
builddentireassembly	isfulfillable	projectexpensetype	storedisplaythumbnail
class	isinactive	projecttemplate	totalvalue
copydescription	islotitem	purchasedescription	tracklandedcost
cost	isonline	purchaseorderamount	transferprice
costestimate	isserialitem	purchaseorderquantity	unbuildvarianceaccount
costestimatetype	isspecialorderitem	purchaseorderquantitydiff	unitstype
costingmethod	isspecialworkorderitem	purchaseunit	upccode
countryofmanufacture	itemid	quantitypricingschedule	urlcomponent
createddate	itemprocessfamily	rate	usecomponentyield
createjob	itemprocessgroup	receiptamount	usemarginalrates
custreturnvarianceaccount	itemtype	receiptquantity	vendor
deferralaccount	lastpurchaseprice	receiptquantitydiff	vendorname
deferredrevenueaccount	leadtime	relateditemsdescription	vendreturnvarianceaccount
deferrevrec	location	reordermultiple	vsoedeferral
demandmodifier	manufacturer	reorderpoint	vsoedelivered
department	matchbilltoreceipt	residual	vsoepermtdiscount
description	matrixitemnametemplate	revrecschedule	vsoeprice
displayname	maxdonationamount	roundupascomponent	vsoesopgroup
dontshowprice	maximumquantity	safetystocklevel	weight
dropshipexpenseaccount	metataghtml	safetystockleveldays	weightunit
effectivebomcontrol	minimumquantity	saleunit	

Synthetic Fields			
commercecATEGORY	ispurchasable	itemoptions_detail	quantityonhand_detail
onlinecustomerprice_detail	correlateditems	relateditems	quantityonorder_detail
isbackorderable	defaultitemshipmethod	quantityavailable_detail	showoutofstockmessage
isdisplayable	itemimages_detail	quantitybackordered_detail	
isinstock	matrixchilditems	quantitycommitted_detail	

Errors

Error Code	Thrown If
SSS_INVALID_TYPE_ARG	Parameter is invalid
FIELD_1_CANNOT_BE_EMPTY	Required parameter is missing or empty

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [Ncommerce/recordView Script Sample](#).

```

1 // Add additional code here
2 ...
3 try {
4     result.viewItems = recordView.viewItems({
5         ids: [408],
6         fields: ["itemtype", "isavailable"]
7         fieldOptions: [{"includeVat":true}]
8     });
9 }
10 catch (e) {
11     result.error = e.name + ": " + e.message;
12 }
13 ...

```

Returns

Returns a flat JSON structure with field:value pairs.

```

1 [
2     {
3         "internalid": {
4             value : 523
5         },
6         "name": {
7             value : "test"
8         },
9         "dropdownListField": {
10            value : {
11                "id": 1,
12                "label":"Option 1"
13            }
14        },
15        "checkbox1Field": {
16            value : true
17        },
18        "dateField1Field": {
19            value : "2012-04-23T18:25:43.511Z"
20        }
21    }, ...
22 ]

```

recordView.viewWebsite(options)

Applies to: Commerce Web Stores



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves the website details with requested website fields.
Returns	See the Returns section.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	Ncommerce/recordView Module
Sibling Module Members	Ncommerce/recordView Module Members
Since	2019.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Property	Type	Required / Optional	Description
options.id	number	required	ID of the website
options.fields	string string[]	required	Website fields you want to retrieve. For more information, see Supported Fields .
options.fieldOptions	Array of name, value pairs. Type depends upon parameter.	optional	Options that affect all related fields that are retrieved. Supported field options: None

Supported Fields

The following fields can be requested in the options.fields parameter:

Record Fields			
analyticsclickattributes	displaycompanyfield	onlinepricelevel	shipstoallcountries
analyticssubmitattributes	displayname	outofstockbehavior	showcookieconsentbanner
autoapplypromotionsenabled	hidepaymentpagewhennobalance	outofstockitems	showpofieldonpayment
cartsharingmode	igniteedition	requestshippingaddressfirst	showextendedcart
confpagetrackinghtml	includevatwithprices	requirecompanyfield	showsavreditinfo
cookiepolicy	isinactive	requireloginforpricing	showshippingestimator
customeregistrationtype	iswebstoreoffline	requireshippinginformation	siteloginrequired
defaultshippingcountry	internalid	requiretermsandconditions	subsidiaries
defaultshippingmethod	loginallowed	savecreditinfo	termsandconditionshtml

Synthetic Fields			
currencies	imagesizes	legacytouchpoints	sortfields
facetfields	languages	shiptocountries	subsidiaries

Errors

Error Code	Thrown If
SSS_INVALID_TYPE_ARG	Parameter is invalid
FIELD_1_CANNOT_BE_EMPTY	Required parameter is empty

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [Ncommerce/recordView Script Sample](#).

```

1 // Add additional code here
2 ...
3 try {
4     result.viewItems = recordView.viewWebsite({
5         id: 2,
6         fields: ["shiptocountries"]
7     });
8 }
9 catch (e) {
10     result.error = e.name + ":" + e.message;
11 }
12 ...

```

Returns

Returns a flat JSON structure with field:value pairs.

```

1 [
2     {
3         "internalid": {
4             value : 523
5         },
6         "name": {
7             value : "test"
8         },
9         "dropdownListField": {
10             value : {
11                 "id": 1,
12                 "label": "Option 1"
13             }
14         },
15         "checkbox1Field": {
16             value : true
17         },
18         "dateField1Field": {
19             value : "2012-04-23T18:25:43.511Z"
20         }
21     }, ...
22 ]

```

N/compress Module

Load the N/compress module to compress and decompress files. You can also use these APIs to archive multiple files in a single file archive such as TAR or ZIP file.

You can compress and decompress individual files by using [compress.gzip\(options\)](#) and [compress.gunzip\(options\)](#).

You can create an archive by using [compress.createArchiver\(\)](#) and add multiple files to the archive.

N/compress Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	compress.Archiver	Object	Server scripts	The functionality for creating archive files.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				Use compress.createArchiver() to create this object.
Method	compress.createArchiver()	compress.Archiver	Server scripts	Creates a compress.Archiver object.
	compress.gunzip(options)	file.File	Server scripts	Decompresses a file and returns it as a temporary file object.
	compress.gzip(options)	file.File	Server scripts	Compresses a file and returns it as a temporary file object.
Enum	compress.Type	enum	Server scripts	Holds the string values for the archive types.

Archiver Object Members

The following members are called on the [compress.Archiver](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	Archiver.add(options)	void	Server scripts	Adds a file to be archived.
	Archiver.archive(options)	file.File	Server scripts	Creates an archive with the added files and returns it as a temporary file object.

N/compress Module Script Samples

Note: This sample script uses the `require` function so you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (a script you attach to a script record and deploy). For more information, see the help topic [SuiteScript 2.0 Script Basics](#) SuiteScript 2.0 Script Basics and [SuiteScript 2.0 Script Types](#) SuiteScript 2.0 Script Types.

The following sample compresses and decompresses a file.

```

1 require(['N/compress', 'N/file'], function(compress, file) {
2     var unsavedTxtFile = file.create({
3         fileType: 'PLAINTEXT',
4         name: 'file.txt',
5         contents: 'This is a sample content. This is a sample content. This is a sample content. This is only a sample.'
6     });
7
8     log.debug('#### ORIGINAL FILE #### #');
9     log.debug('Name: ' + unsavedTxtFile.name);
10    log.debug('Size: ' + unsavedTxtFile.size + 'b');
11    log.debug('Contents: ' + unsavedTxtFile.getContents());
12
13    log.debug('#### GZIPPED FILE WITH MAX COMPRESSION ####');
14    var gzippedFile = compress.gzip({
15        file: unsavedTxtFile,
16        level: 9
17    });
18    log.debug('Name: ' + gzippedFile.name);
19    log.debug('Size: ' + gzippedFile.size + 'b');
20    log.debug('Contents: ' + gzippedFile.getContents().substring(0, 100));
21
22    log.debug('#### GUNZIPPED FILE ####');

```

```

23 | var gunzippedFile = compress.gunzip({
24 |   file: gzippedFile
25 | });
26 | log.debug('Name: ' + gunzippedFile.name);
27 | log.debug('File size: ' + gunzippedFile.size + 'b');
28 | log.debug('Contents: ' + gunzippedFile.getContents());
29 |
30 });
31 });

```

Note: This sample script uses the require function so you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (a script you attach to a script record and deploy). For more information, see the help topic [SuiteScript 2.0 Script Basics](#) SuiteScript 2.0 Script Basics and [SuiteScript 2.0 Script Types](#) SuiteScript 2.0 Script Types.

The following sample creates a ZIP file.

```

1 | require(['N/compress', 'N/file'], function(compress, file) {
2 |   // load/create files to be archived
3 |   var binaryFile = file.load({
4 |     id: 200
5 |   });
6 |   var textFile = file.create({
7 |     name: 'file.txt',
8 |     fileType: 'PLAINTEXT',
9 |     contents: 'This is just a sample content.'
10 |   });
11 |
12 |   // create an archive as a temporary file object
13 |   var archiver = compress.createArchiver();
14 |   archiver.add({
15 |     file: binaryFile
16 |   });
17 |   archiver.add({
18 |     file: textFile,
19 |     directory: 'txt/'
20 |   });
21 |   var zipFile = archiver.archive({
22 |     name: 'myarchive.zip'
23 |   });
24 |
25 |   // save the archive to file cabinet
26 |   zipFile.folder = 123;
27 |   zipFile.save();
28 | });

```

compress.Archiver

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The functionality for creating an archive file. Use compress.createArchiver() to create this object.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/compress Module
Methods and Properties	Archiver Object Members
Since	2020.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/compress Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var archiver = compress.createArchiver();
4 ...
5 // Add additional code

```

Archiver.add(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a file to be archived. In the archive, the path to the target file is options.directory or options.file.name. If the options.directory parameter is not specified, the target file is located in the root directory of the archive.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/compress Module
Parent Object	compress.Archiver
Sibling Object Members	Archiver Object Members
Since	2020.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.file	file.File	required	The file to be archived.
options.directory	string	optional	The target directory in the archive. If this parameter is not specified, the file is placed in the root directory of the archive.

Errors

Error Code	Thrown If
COMPRESS_API_DUPLICATE_PATH	A file has already been added using the same target path.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/compress Module Script Samples](#).

```

1 // Add additional code
2 ...
3 archiver.add({
4   file: myJpegFile,
5   directory: 'pics/'
6 });
7 archiver.add({
8   file: myTxtFile
9 });
10 ...
11 // Add additional code

```

Archiver.archive(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates an archive with the added files and returns it as a temporary file object.
Returns	file.File
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/compress Module
Parent Object	compress.Archiver
Sibling Object Members	Archiver Object Members
Since	2020.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	required	The name of the archive file.
options.type	string	optional	<p>The archive type. See the compress.Type enum.</p> <p>This parameter does not need to be specified if options.name has one of the following extensions:</p> <ul style="list-style-type: none"> ▪ .cpio ▪ .tar ▪ .tar.gz ▪ .tar.bz2 ▪ .tgz

Parameter	Type	Required / Optional	Description
			<ul style="list-style-type: none"> ■ .tbz2 ■ .zip

Errors

Error Code	Thrown If
COMPRESS_API_UNSUPPORTED_ARCHIVE_TYPE	The options.type parameter is an invalid archive type.
COMPRESS_API_UNRECOGNIZED_ARCHIVE_FILE_EXTENSION	The options.type parameter is not specified and the archive type cannot be determined.
COMPRESS_API_UNABLE_TO_RETRIEVE_FILE_CONTENTS	The contents cannot be retrieved for any of the files to be archived.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/compress Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var archiver = compress.createArchiver();
4 archiver.add({
5   file: binaryFile
6 });
7 archiver.add({
8   file: textFile,
9   directory: 'txt/'
10 });
11
12 var zipFile = archiver.archive({
13   name: 'myarchive.zip'
14 });
15 var tgzFile = archiver.archive({
16   name: 'myarchive.tar.gz'
17 });
18 // Add additional code

```

compress.gzip(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Compresses a file by using gzip and returns it as a temporary file object. 0 is no compression. 9 is the best compression level.
Returns	<code>file.File</code>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/compress Module
Module Members	N/compress Module Members
Since	2020.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.file	file.File	required	The file to be compressed.
options.level	number	optional	The compression level. 0 is no compression. 9 is the best compression level.

Errors

Error Code	Thrown If
COMPRESS_API_UNABLE_TO_RETRIEVE_FILE_CONTENTS	The contents of the file to be compressed cannot be retrieved.
COMPRESS_API_COMPRESSION_LEVEL_OUT_OF_RANGE	The specified compression level is outside of the valid range (0 - 9).

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/compress Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var gzippedFile = compress.gzip({
4   file: myFile,
5   level: 9
6 });

```

compress.gunzip(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Decompresses a file that was compressed using gzip and returns it as a temporary file object.
Returns	file.File
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/compress Module
Module Members	N/compress Module Members
Since	2020.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.file	<code>file.File</code>	required	The file to be decompressed.

Errors

Error Code	Thrown If
COMPRESS_API_DECOMPRESS_ERROR	The file cannot be decompressed.
COMPRESS_API_UNABLE_TO_RETRIEVE_FILE_CONTENTS	The contents of the file to be decompressed cannot be retrieved.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/compress Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var gunzippedFile = compress.gunzip({
4   file: gzippedFile
5 });

```

compress.createArchiver()

Method Description	Creates a <code>compress.Archiver</code> object that can be used for creating file archives, such as ZIP or TAR files.
Returns	<code>compress.Archiver</code>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/compress Module
Module Members	N/compress Module Members
Since	2020.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/compress Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var archiver = compress.createArchiver();
4 ...
5 // Add additional code

```

compress.Type



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for the archive types.
Module	N/compress Module
Sibling Module Members	N/compress Module Members
Since	2020.2

Values

Value
CPIO
TAR
TBZ2
TGZ
ZIP

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/compress Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var tarArchive = archiver.archive({
4   name: 'myarchive',
5   type: compress.Type.TAR
6 });
7 // Add additional code

```

N/config Module

Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/config module when you want to access NetSuite configuration settings. The `config.load(options)` method returns a `record.Record` object. Use the `record.Record` object members to access configuration settings. You do not need to load the record module to do this.

See [config.Type](#) for a list of supported configuration objects.

- [N/config Module Members](#)
- [N/config Module Script Sample](#)

N/config Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	<code>config.load(options)</code>	<code>record.Record</code>	Server-side scripts	Loads a <code>record.Record</code> object that encapsulates the specified configuration page.
Enum	<code>config.Type</code>	enum	Server-side scripts	Holds the string values for supported configuration objects. This enum is used to set the value of the NetSuite configuration page you want to access.

N/config Module Script Sample

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads the Company Information configuration page. It then sets the values specified for the Tax ID Number field and the Employer Identification Number field.

Note: The IDs in this sample are placeholders. Replace the Tax ID Number field and the Employer Identification Number with valid IDs from your NetSuite account.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/config'],
6   function(config) {
7     function setTaxAndEmployerId() {
8       var companyInfo = config.load({
9         type: config.Type.COMPANY_INFORMATION
10      });
11      companyInfo.setValue({
12        fieldId: 'taxid',
13        value: '1122334455'
14      });
15      companyInfo.setValue({
16        fieldId: 'employerid',
17        value: '123456789'
18      });
}

```

```

19 |         companyInfo.save();
20 |         companyInfo = config.load({
21 |             type: config.Type.COMPANY_INFORMATION
22 |         });
23 |         var taxid = companyInfo.getValue({
24 |             fieldId: 'taxid'
25 |         });
26 |     });
27 |     setTaxAndEmployerId();
28 | });

```

config.load(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Method used to load a record.Record object that encapsulates the specified NetSuite configuration page.</p> <p>After the configuration page loads, all preference names and IDs are available to get or set. For more information, see the help topic Preference Names and IDs.</p> <p>You can use the following Record Object Members to get and set preference names and IDs:</p> <ul style="list-style-type: none"> ▪ Record.getField(options) ▪ Record.getFields() ▪ Record.getText(options) ▪ Record.getValue(options) ▪ Record.setText(options) ▪ Record.setValue(options)
Returns	record.Record
Supported Script Types	<p>Server-side scripts</p> <p>For additional information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	10 units
Module	N/config Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	enum	required	The NetSuite configuration page you want to access. Use the config.Type enum to set the value.	2015.2
options.isDynamic	boolean true false	optional	<p>Determines whether the record is loaded in dynamic mode.</p> <ul style="list-style-type: none"> ▪ If set to true, the record is loaded in dynamic mode. 	2015.2

Parameter	Type	Required / Optional	Description	Since
			<ul style="list-style-type: none"> ■ If set to false, the record is loaded in standard mode. <p>For more information, see the help topic SuiteScript 2.0 – Standard and Dynamic Modes.</p>	

Error Code	Message	Thrown If
INVALID_RCRD_TYPE	The record type {type} is invalid.	The type argument is invalid or missing.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/config Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var configRecObj = config.load({
4   type: config.Type.COMPANY_INFORMATION
5 });
6 configRecObj.setText({
7   fieldId: 'fiscalmonth',
8   text: 'July'
9 });
10 configRecObj.save();
11 ...
12 //Add additional code

```

config.Type

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

 **Note:** JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Enumeration that holds the string values for supported configuration pages.
Module	N/config Module
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Since	2015.2

Values

Value	Configuration Page
USER_PREFERENCES	Set Preferences page (Home > Set Preferences) For more information about the fields on the page, see the help topic User Preferences .

Value	Configuration Page
COMPANY_INFORMATION	Company Information page (Setup > Company > Company Information) For more information about the fields on the page, see the help topic Company Information .
COMPANY_PREFERENCES	General Preferences page (Setup > Company > General Preferences) For more information about the fields on the page, see the help topic General Preferences .
ACCOUNTING_PREFERENCES	Accounting Preferences page (Setup > Accounting > Accounting Preferences) For more information about the fields on the page, see the help topic Accounting Preferences .
ACCOUNTING_PERIODS	Accounting Periods page (Setup > Accounting > Manage Accounting Periods) For more information about the fields on the page, see the help topic Accounting Periods .
TAX_PERIODS	Tax Periods page (Setup > Accounting > Manage Tax Periods) For more information about the fields on the page, see the help topic Tax Periods .
FEATURES	Enable Features page (Setup > Company > Enable Features) For more information about feature names and IDs, see the help topic Feature Names and IDs .
TIME_POST	For additional information, see the help topic Posting Time Transactions .
TIME_VOID	For additional information, see the help topic Posting Time Transactions .

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/config Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var configRecObj = config.load({
4   type: config.Type.COMPANY_INFORMATION
5 });
6 configRecObj.setText({
7   fieldId: 'fiscalmonth',
8   text: 'July'
9 });
10 configRecObj.save();
11 ...
12 //Add additional code

```

N/crypto Module

i Note: The content in this help topic pertains to SuiteScript 2.0.

The N/crypto module encapsulates hashing, hash-based message authentication (hmac), and symmetrical encryption.

When the crypto module is used, SuiteScript also loads [N/encode Module](#).

- [N/crypto Module Members](#)
- [Cipher Object Members](#)
- [CipherPayload Object Members](#)
- [Decipher Object Members](#)
- [Hash Object Members](#)
- [Hmac Object Members](#)
- [SecretKey Object Members](#)
- [N/crypto Module Script Samples](#)

N/crypto Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	crypto.Cipher	Object	Server-side scripts	Encapsulates a cipher.
	crypto.CipherPayload	Object	Server-side scripts	Encapsulates a cipher payload.
	crypto.Decipher	Object	Server-side scripts	Encapsulates a decipher.
	crypto.Hash	Object	Server-side scripts	Encapsulates a hash.
	crypto.Hmac	Object	Server-side scripts	Encapsulates an hmac.
	crypto.SecretKey	Object	Server-side scripts	Encapsulates a secret key handle.
Method	crypto.createCipher(options)	Object	Server-side scripts	Creates and returns a new crypto.Cipher Object.
	crypto.createDecipher(options)	Object	Server-side scripts	Creates and returns a new crypto.Decipher object.
	crypto.createHash(options)	Object	Server-side scripts	Creates and returns a new crypto.Hash Object.
	crypto.createHmac(options)	Object	Server-side scripts	Creates and returns a new crypto.Hmac Object.
	crypto.createSecretKey(options)	Object	Server-side scripts	Creates and returns a new crypto.SecretKey Object.
Enum	crypto.EncryptionAlg	string (read-only)	Server-side scripts	Holds the string values for supported encryption algorithms. Sets the options.algorithm parameter for crypto.createCipher(options) .
	crypto.HashAlg	string (read-only)	Server-side scripts	Holds the string values for supported hashing algorithms. Sets the value of the options.algorithm parameter for crypto.createHash(options) and crypto.createHmac(options) .
	crypto.Padding	string (read-only)	Server-side scripts	Holds the string values for supported cipher padding. Sets the options.padding parameter for crypto.createCipher(options) and crypto.createDecipher(options) .

Cipher Object Members

The following members are called on [crypto.Cipher](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Cipher.update(options)	Object	Server-side scripts	Updates the clear data with the specified encoding.
	Cipher.final(options)	Object	Server-side scripts	Returns the cipher data.

CipherPayload Object Members

The following members are called on [crypto.CipherPayload](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	CipherPayload.ciphertext	string	Server-side scripts	The result of the ciphering process.
	CipherPayload.iv	number	Server-side scripts	An initialization vector

Decipher Object Members

The following members are called on [crypto.Decipher](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Decipher.final(options)	string	Server-side scripts	Returns the clear data.
	Decipher.update(options)	void	Server-side scripts	Updates cipher data with the specified encoding.

Hash Object Members

The following members are called on [crypto.Hash](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Hash.digest(options)	string	Server-side scripts	Calculates the digest of the data to be hashed.
	Hash.update(options)	void	Server-side scripts	Updates the clear data with the encoding specified.

Hmac Object Members

The following members are called on [crypto.Hmac](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Hmac.digest(options)	string	Server-side scripts	Gets the computed digest.
	Hmac.update(options)	void	Server-side scripts	Updates the clear data with the encoding specified.

SecretKey Object Members

The following members are called on `crypto.SecretKey`.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	<code>Secretkey.guid</code>	string	Server-side scripts	The GUID associated with the secret key.
	<code>SecretKey.encoding</code>	string	Server-side scripts	The encoding used for the clear text value of the secret key.

N/crypto Module Script Samples

The following script samples demonstrate how to use the features of the N/crypto module.

Sample 1: Generate a secure key

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample demonstrates the APIs needed to generate a secure key using the SHA512 hashing algorithm. The GUID in this sample is a placeholder. You must replace it with a valid value from your NetSuite account. To create a real password GUID, obtain a password value from a credential field on a form. For more information, see [Form.addCredentialField\(options\)](#). Also see [N/https Module Script Sample](#) for a Suitelet sample that shows creating a form field that generates a GUID.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/crypto', 'N/encode', 'N/runtime'], function(crypto, encode, runtime) {
6    function createSecureKeyWithHash() {
7      var inputString = 'YWJjZGVmZwo=';
8      var myGuid = '{284CFB2D225B1D76FB94D150207E49DF}';
9
10     var sKey = crypto.createSecretKey({
11       guid: myGuid,
12       encoding: encode.Encoding.UTF_8
13     });
14
15     var hmacSHA512 = crypto.createHmac({
16       algorithm: crypto.HashAlg.SHA512,
17       key: sKey
18     });
19     hmacSHA512.update({
20       input: inputString,
21       inputEncoding: encode.Encoding.BASE_64
22     });
23     var digestSHA512 = hmacSHA512.digest({
24       outputEncoding: encode.Encoding.HEX
25     });
26   }
27
28   createSecureKeyWithHash();
29 });

```

Sample 2: Create a Suitelet to work with keys



Note: This script sample uses the `define` function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the `require` function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample creates a simple Suitelet that requests user credentials, creates a secret key, and encodes a sample string.



Note: The default maximum length for a secret key field is 32 characters. If needed, use the `Field.maxLength` property to change this value.

```

1  /**
2   * @NApiVersion 2.x
3   * @NScriptType Suitelet
4   */
5 define(['N/ui/serverWidget', 'N/runtime', 'N/crypto', 'N/encode'], function(ui, runtime, crypto, encode) {
6     function onRequest(option) {
7       if (option.request.method === 'GET') {
8         var form = ui.createForm({
9           title: 'My Credential Form'
10        });
11
12        var skField = form.addSecretKeyField({
13          id: 'mycredential',
14          label: 'Credential',
15          restrictToScriptIds: [runtime.getCurrentScript().id],
16          restrictToCurrentUser: false
17        })
18        skField.maxLength = 200;
19
20        form.addSubmitButton();
21        option.response.writePage(form);
22      } else {
23        var form = ui.createForm({
24          title: 'My Credential Form'
25        });
26
27        var inputString = "YWJjZGVmZwo=";
28        var myGuid = option.request.parameters.mycredential;
29
30        // Create the key
31        var sKey = crypto.createSecretKey({
32          guid: myGuid,
33          encoding: encode.Encoding.UTF_8
34        });
35
36        try {
37          var hmacSha512 = crypto.createHmac({
38            algorithm: 'SHA512',
39            key: sKey
40          });
41          hmacSha512.update({
42            input: inputString,
43            inputEncoding: encode.Encoding.BASE_64
44          });
45          var digestSha512 = hmacSha512.digest({
46            outputEncoding: encode.Encoding.HEX
47          });
48        } catch (e) {
49          log.error({
50            title: 'Failed to hash input',
51            details: e
52          });
53        }
54      }
    }
  
```

```

55     form.addField({
56         id: 'result',
57         label: 'Your digested hash value',
58         type: 'textarea'
59     }).defaultValue = digestSha512;
60
61     option.response.writePage(form);
62 }
63
64 return {
65     onRequest: onRequest
66 };
67 });

```

crypto.Cipher



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a cipher. For a complete list of this object's methods and properties, see Cipher Object Members .
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var cipher = crypto.createCipher({
4     algorithm: crypto.EncryptionAlg.AES,
5     key: sKey
6 });
7 ...
8 //Add additional code

```

Cipher.final(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to return the cipher data. Sets the output encoding for the crypto.CipherPayload object.
Returns	A crypto.CipherPayload Object
Supported Script Types	Server-side scripts

	For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.outputEncoding	enum	optional	<p>The output encoding for a crypto.CipherPayload object.</p> <p>The default value is HEX.</p> <p>Use the encode.Encoding enum to set the value.</p>

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto Module Script Sample](#).

```

1 //Add additional code
2 ...
3 crypto.createCipher({
4   algorithm: crypto.EncryptionAlg.AES,
5   key: sKey
6 });
7
8 var cipherPayload = cipher.final({
9   outputEncoding: encode.Encoding.BASE_64
10 });
11 ...
12 //Add additional code

```

Cipher.update(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to update the clear data with the specified encoding.
Returns	Void
Supported Script Types	<p>Server-side scripts</p> <p>For additional information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.input	string	required	The clear data to be updated.
options.inputEncoding	enum	optional	The input encoding. Use the encode.Encoding enum to set the value. The default value is UTF_8.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var reencoded = Cipher.update({
4   input: 'Carrot cake gummi bears'
5 });
6 ...
7 //Add additional code

```

crypto.CipherPayload

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a cipher payload. For a complete list of this object's methods and properties, see CipherPayload Object Members .
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 crypto.createCipher({
4   algorithm: crypto.EncryptionAlg.AES,

```

```

5   key: sKey
6 });
7
8 var cipherPayload = cipher.final({
9   outputEncoding: encode.Encoding.HEX
10 });
11 ...
12 //Add additional code

```

CipherPayload.ciphertext



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The result of the ciphering process. For example, to take the cipher payload and send it to another system.
Type	string
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto Module Script Sample](#).

```

1 //Add additional code
2 ...
3 log.debug({
4   title: "Ciphertext: ",
5   details: cipherPayload.ciphertext
6 });
7 ...
8 //Add additional code

```

CipherPayload.iv



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Initialization vector for the cipher payload. You can pass in the iv value to crypto.createDecipher(options)
Type	string
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module

Since	2015.2
--------------	--------

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 log.debug({
4   title: "CipherPayload IV: ",
5   details: cipherPayload.iv
6 });
7 ...
8 //Add additional code

```

crypto.Decipher



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a decipher. This object has methods that decrypt. For a complete list of this object's methods and properties, see Decipher Object Members .
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 crypto.createDecipher({
4   algorithm: crypto.EncryptionAlg.AES,
5   key: sKey
6 });
7 ...
8 //Add additional code

```

Decipher.final(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to return the clear data.
---------------------------	---------------------------------------

Returns	string
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.outputEncoding	string	optional	Specifies the encoding for the output Set the value using the encode.Encoding enum. The default value is UTF_8.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var decipher1 = Decipher.final({
4   outputEncoding: encode.Encoding.HEX
5 });
6 ...
7 //Add additional code

```

Decipher.update(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to update cipher data with the specified encoding.
Returns	Void
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.input	string	required	The data to update
options.inputEncoding	string	optional	Specifies the encoding of the input data Set the value using the encode.Encoding enum. The default value is HEX.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var decipher1 = Decipher.update({
4   input: '73616d706c65737472696e67',
5   inputEncoding: encode.Encoding.HEX
6 });
7 ...
8 //Add additional code

```

crypto.Hash

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a hash. For a complete list of this object's methods and properties, see Hash Object Members .
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var hashObj = crypto.createHash({
4   algorithm: crypto.HashAlgorithm.SHA256
5 });
6 ...
7 //Add additional code

```

Hash.digest(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Calculates the digest of the data to be hashed.
Returns	A hash value as a string
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.outputEncoding	string	optional	The output encoding. Set using the encode.Encoding enum. The default value is HEX.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var digestSample = hashObj.digest({
4   outputEncoding: encode.Encoding.HEX
5 });
6 ...
7 //Add additional code
  
```

Hash.update(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to update clear data with the encoding specified.
Returns	Void
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.input	string	required	The data to be updated.
options.inputEncoding	string	optional	The input encoding. Set using the encode.Encoding enum. The default value is UTF_8.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var inputString = 'Lemon drops ice cream jelly marzipan cake';
4 hashSample.update({
5   input: inputString
6 });
7 ...
8 //Add additional code

```

crypto.Hmac

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates an hmac. For a complete list of this object's methods and properties, see Hmac Object Members .
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var hmacSHA512 = crypto.createHmac({
4   algorithm: crypto.HashAlg.SHA512,
5   key: sKey
6 });

```

```

7 | ...
8 | //Add additional code

```

Hmac.digest(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the computed digest.
Returns	An hmac value as a string
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.outputEncoding	string	optional	Specifies the encoding of the output string. Set using the encode.Encoding enum. The default value is HEX.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var digestSHA512 = hmacSHA512.digest({
4   outputEncoding: encode.Encoding.HEX
5 });
6 ...
7 //Add additional code

```

Hmac.update(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to update the clear data with the encoding specified.
Returns	Void
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.input	string	required	The hmac data to be updated.
options.inputEncoding	enum	optional	The input encoding. Set using the encode.Encoding enum. The default value is UTF_8.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 hmacSHAS12.update({
4     input: inputString,
5     inputEncoding: encode.Encoding.BASE_64
6 });
7 ...
8 //Add additional code

```

crypto.SecretKey

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the handle to the key. The handler does not store the key value. It points to the key stored within the NetSuite system. The GUID is also required to find the key. For a complete list of this object's methods and properties, see SecretKey Object Members .
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```
1 //Add additional code
```

```

2 ...
3 var sKey = crypto.createSecretKey({
4   guid: '284CFB2D225B1D76FB94D150207E49DF',
5   encoding: encode.Encoding.UTF_8
6 });
7 ...
8
9 //Add additional code

```

SecretKey.encoding



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The encoding used for the clear text value of the secret key.
Type	string
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto Module Script Sample](#).

```

1 //Add additional code
2 ...
3 log.debug({
4   title: "Secret Key Encoding: ",
5   details: sKey.encoding
6 });
7 ...
8 //Add additional code

```

Secretkey.guid



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The GUID associated with the secret key.
Type	string
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 log.debug({
4   title: "Secret Key GUID: ",
5   details: sKey.guid
6 });
7 ...
8 //Add additional code

```

crypto.createCipher(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to create and return a crypto.EncryptionAlg object.
	Note: The blockCipherMode is automatically set to CBC.
Returns	A crypto.EncryptionAlg object
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.algorithm	string	required	The hash algorithm. Set the value using the crypto.EncryptionAlg enum.	2015.2
options.key	object	required	The crypto.SecretKey object.	2015.2
			Note: When using the crypto.SecretKey object for an AES algorithm, the length of the text (secret key) that is used to generate the GUID must be 16, 24, or 32 characters.	
options.padding	string	optional	The padding for the cipher text. Set the value using the crypto.Padding enum. By default, the value is set to PKCS5Padding.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var cipher = crypto.createCipher({
4   algorithm: crypto.EncryptionAlg.AES,
5   key: sKey,
6   padding: crypto.Padding.PKCS5Padding
7 });
8 ...
9 //Add additional code

```

crypto.createDecipher(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to create a crypto.Decipher object.
	i Note: The blockCipherMode is automatically set to CBC.
Returns	A crypto.Decipher object.
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.algorithm	string	required	The hash algorithm. Set by the crypto.EncryptionAlg enum.	2015.2
options.key	object	required	The crypto.SecretKey object used for encryption. i Note: When using the crypto.SecretKey object for an AES algorithm, the length of the text (secret key) that is used to generate the GUID must be 16, 24, or 32 characters.	2015.2
options.padding	object	optional	The padding for the cipher. Set the value using the crypto.Padding enum.	2015.2

Parameter	Type	Required / Optional	Description	Since
options.iv	string	required	The initialization vector that was used for encryption.	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var decipher = crypto.createDecipher({
4   algorithm: crypto.EncryptionAlg.AES,
5   key: sKey,
6   padding: NoPadding,
7   iv: '2311141720'
8 });
9 ...
10 //Add additional code

```

crypto.createHash(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to create a crypto.Hash object.
Returns	The crypto.Hash object created using this method.
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.algorithm	string	required	■ The hash algorithm. Set using the crypto.HashAlg enum.	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```
1 //Add additional code
```

```

2 ...
3 var hashObj = crypto.createHash({
4   algorithm: crypto.HashAlg.SHA256
5 });
6 ...
7 //Add additional code

```

crypto.createHmac(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to create a crypto.Hmac object.
Returns	A crypto.Hmac object.
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.algorithm	string	required	The hash algorithm. Use the crypto.HashAlg enum to set this value.	2015.2
options.key	object	required	The crypto.SecretKey object.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var hmacObj = crypto.createHmac({
4   algorithm: HashAlg.SHA256,
5   key: sKey
6 });
7 ...
8 //Add additional code

```

crypto.createSecretKey(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to create a new crypto.SecretKey object.
---------------------------	--

	This method can take a GUID. Use Form.addCredentialField(options) to generate a value.
	<p>Note: When using the crypto.SecretKey object for an AES algorithm, the length of the text (secret key) that is used to generate the GUID must be 16, 24, or 32 characters.</p>
Returns	A crypto.SecretKey object
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.guid	string	required	A GUID used to generate a secret key. The GUID can resolve to either data or metadata.	2015.2
options.encoding	enum	optional	Specifies the encoding for the SecureKey. Set this value using the encode.Encoding enum. The default value is HEX.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var secretKey = crypto.createSecretKey({
4   encoding: encode.Encoding.HEX,
5   guid: '284CFB2D225B1D76FB94D150207E49DF'
6 });
7 ...
8 //Add additional code

```

crypto.EncryptionAlg

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported encryption algorithms. Sets the options.algorithm parameter for crypto.createCipher(options) .
-------------------------	--

	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/crypto Module
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Since	2015.2

Values

- AES

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var cipher = crypto.createCipher({
4   algorithm: crypto.EncryptionAlg.AES,
5   key: sKey
6 });
7 ...
8 //Add additional code

```

crypto.HashAlg

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported hashing algorithms. Sets the value of the options.algorithm parameter for crypto.createHash(options) and crypto.createHmac(options) .
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/crypto Module
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Since	2015.2

Values

- SHA1

- SHA256
- SHA512
- MD5

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var hmacSHA512 = crypto.createHmac({
4   algorithm: crypto.HashAlg.SHA512,
5   key: sKey
6 });
7 ...
8 //Add additional code

```

crypto.Padding



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported cipher padding. Sets the options.padding parameter for crypto.createCipher(options) and crypto.createDecipher(options) .
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/crypto Module
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Since	2015.2

Values

- NoPadding
- PKCS5Padding

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/crypto Module Script Sample.

```

1 //Add additional code
2 ...
3 var cipher = crypto.createCipher({
4   algorithm: crypto.EncryptionAlg.AES,
5   key: sKey,

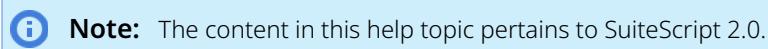
```

```

6   padding: crypto.Padding.NoPadding
7 });
8 ...
9 //Add additional code

```

N/crypto/certificate Module



Load the N/crypto/certificate module to sign XML documents or strings with digital certificates using asymmetric cryptography. In addition to signing XML documents, you can create signer and verifier objects and verify signed documents with this module.

The N/crypto/certificate module includes:

- [N/crypto/certificate Module Members](#)
- [Signer Object Members](#)
- [SignedXml Object Members](#)
- [Verifier Object Members](#)
- [N/crypto/certificate Module Script Samples](#)

N/crypto/certificate Module Members

Member Type	Name	Return Type/ Value Type	Supported Script Types	Description
Object	certificate.SignedXml	Object	Server scripts	Object for an XML string that has been digitally signed. Use certificate.signXml(options) to create this object.
	certificate.Signer	Object	Server scripts	Object for creating signatures for plain strings. Use certificate.createSigner(options) to create this object.
	certificate.Verifier	Object	Server scripts	Object for verifying plain string signatures. Use certificate.createVerifier(options) to create this object.
Method	certificate.verifyXmlSignature(options)	void	Server scripts	Verifies the signature in the SignedXml.asFile() file.
	certificate.createSigner(options)	certificate.Signer	Server scripts	Creates a signer object for signing plain strings.
	certificate.createVerifier(options)	certificate.Verifier	Server scripts	Creates a verifier object for verifying signatures of plain strings.
	certificate.signXml(options)	certificate.SignedXml	Server scripts	Signs the input XML string using the Certificate ID. Returns the SignedXml as a string.
Enum	certificate.HashAlg	enum	Server scripts	Holds the values for hash algorithms.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				SHA1, SHA256, SHA384, or SHA512 are supported digest methods and values for this enum.

Signer Object Members

The following members are called on the [certificate.Signer](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Signer.sign(options)	void	Server scripts	Signs the string and returns the signature.
	Signer.update(options)	void	Server scripts	Updates the input string to be signed. The string can be encoded.

Verifier Object Members

The following members are called on the [certificate.Verifier](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Verifier.update(options)	void	Server scripts	Updates the string to be verified against specified certificate.
	Verifier.verify(options)	void	Server scripts	Verifies the string against provided signature using specified certificate.

SignedXml Object Members

The following members are called on the [certificate.SignedXml](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	SignedXml.asFile()	file.File	Server scripts	Returns the signed XML as a file object.
	SignedXmlasString()	string	Server scripts	Returns the signed XML as a string.
	SignedXml.asXml()	xml.Document	Server scripts	Returns the signed XML as an XML document. You can use the N/xml Module with this document to access elements and attributes in the XML.

N/crypto/certificate Module Script Samples

The following script samples demonstrate how to use the features of the N/crypto/certificate module.

Sample 1: Load and Sign XML File

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads an XML file from the File Cabinet and signs it using the digital certificate with internal ID 'custcertificate1'.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/crypto/certificate', 'N/file'],
6   function (cert, file){
7     var infNFe = file.load({
8       id: 922
9     });
10    var signedXml = cert.signXML({
11      algorithm: 'SHA1',
12      certId: 'custcertificate1',
13      rootTag: 'infNFe',
14      xmlString: infNFe.getContents()
15    });
16    cert.verifyXMLSignature({
17      signedXml:signedXml,
18      rootTag: 'infNFe'
19    });
20  });

```

Sample 2: Create Signer and Verifier Objects

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a `certificate.Signer` object, signs it, and then creates a `certificate.Verifier` object and verifies the signer object.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/crypto/certificate'], function (certificate) {
6   var signer = certificate.createSigner({
7     certId: 'custcertificate1',
8     algorithm: 'SHA1'
9   });
10  signer.update('test');
11  var result = signer.sign();
12  var verifier = certificate.createVerifier({
13    certId: 'custcertificate1',
14    algorithm: 'SHA1'
15  });
16  verifier.update('test');

```

```

17     verifier.verify(result);
18 })

```

certificate.SignedXml



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A signed XML string. This object is returned by the certificate.signXml(options) method.
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto/certificate Module
Methods and Properties	SignedXml Object Members
Since	2019.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto/certificate Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var signedXml = cert.signXML({
4     algorithm: 'SHA1',
5     certId: 'custcertificate1',
6     rootTag: 'infNFe',
7     xmlString: infNFe.getContents()
8 });
9 certificate.verifyXMLSignature({
10     signedXml:signedXml,
11     rootTag: 'infNFe'
12 });
13 ...
14 //Add additional code
15

```

SignedXml.asFile()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the signed XML as a file.
Returns	file.File
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto/certificate Module

Parent Object	certificate.SignedXml
Sibling Object Members	SignedXml Object Members
Since	2019.2

SignedXml.asString()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the signed XML as a string.
Returns	string
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto/certificate Module
Parent Object	certificate.SignedXml
Sibling Object Members	SignedXml Object Members
Since	2019.1

SignedXml.asXml()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the signed XML as an XML document.
Returns	xml.Document
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto/certificate Module
Parent Object	certificate.SignedXml
Sibling Object Members	SignedXml Object Members
Since	2019.2

certificate.Signer

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	Object used for signing plain strings.
---------------------------	--

	This object is returned by the certificate.createSigner(options) method.
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto/certificate Module
Methods and Properties	Signer Object Members
Since	2019.1

Signer.update(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Updates the input string to be signed. The string can be encoded.
Returns	void
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto/certificate Module
Parent Object	certificate.Signer
Sibling Object Members	Signer Object Members
Since	2019.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.input	string	required	The string to update.
options.inputEncoding	string	optional	Encoding of the string to sign (e.g., UTF-8, ISO_8859_1, ASCII). The default value is UTF-8.

 **Note:** This must be a text value. Values from encode.Encoding (N/encode module) are not accepted.

Errors

Error Code	Thrown If
SSS_UNSUPPORTED_ENCODING	The value for is invalid. This must be a text value, such as UTF-8, ISO_8859_1, ASCII. Values from encode.Encoding (N/encode module) are not valid.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto/certificate Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var signer = certificate.createSigner({
4   certId: 'custcertificate1',
5   algorithm: 'SHA1'
6 });
7 signer.update("test");
8 var result = signer.sign();
9 ...
10 //Add additional code

```

Signer.sign(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Signs the string and returns the signature.
Returns	string
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto/certificate Module
Parent Object	certificate.Signer
Sibling Object Members	Signer Object Members
Since	2019.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.outputEncoding	string	optional	Encoding of the signed string in Base64 format.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto/certificate Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var signer = certificate.createSigner({
4   certId: 'custcertificate1',
5   algorithm: 'SHA1'

```

```

6   });
7   signer.update("test");
8   var result = signer.sign();
9   ...
10 //Add additional code

```

certificate.Verifier



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Object for verifying plain string signatures. This object is returned by the certificate.createVerifier(options) method.
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/crypto/certificate Module
Methods and Properties	Verifier Object Members
Since	2019.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/crypto/certificate Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var verifier = certificate.createVerifier({
4     certId: 'custcertificate1',
5     algorithm: 'SHA1'
6 });
7 verifier.update('test');
8 verifier.verify(result);
9 ...
10 //Add additional code

```

Verifier.update(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Updates the string to be verified against a specified certificate.
Returns	void
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto/certificate Module
Parent Object	Parameters

Sibling Object Members	Verifier Object Members
Since	2019.1

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description
options.input	string	required	The string to verify.
options.inputEncoding	string	optional	Encoding of the string to verify. The default value is UTF-8.

Verifier.verify(options)

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Method Description	Verifies a string against a provided signature using a specified certificate. You can create a verifier object using the certificate.createVerifier(options) method.
Returns	void
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/crypto/certificate Module
Parent Object	Parameters
Sibling Object Members	Verifier Object Members
Since	2019.1

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description
options.signature	string	required	The signature to be verified.
options.signatureEncoding	string	optional	The signature's encoding in Base64 format.

Errors

Error Code	Thrown If
INVALID_SIGNATURE	Signature is not verified. This can occur if the certificate or hash algorithm is not correct in the Verifier object or the signature is not valid for the supplied string.

certificate.createSigner(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates the signer object for signing plain strings.
Returns	certificate.Signer
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/crypto/certificate Module
Since	2019.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.certId	string	required	The script ID of the digital certificate.
options.algorithm	string	required	The hash algorithm.

certificate.createVerifier(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates the verifier object for verifying signatures of plain strings.
Returns	A Parameters object
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/crypto/certificate Module
Since	2019.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.certId	string	required	The script ID of the digital certificate.
options.algorithm	string	required	Hash algorithm

certificate.verifyXmlSignature(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Verifies the signature in the signedXml object or string.
Returns	void
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/crypto/certificate Module
Since	2019.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.signedXml	string	required	Signed XML
options.rootTag	string	required	Signed root XML tag.
options.certId	string	optional	The script ID for the digital certificate.

certificate.signXml(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Signs the inputXml string using the certId.
Returns	certificate.SignedXml
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/crypto/certificate Module
Since	2019.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.xmlString	string	required	Input XML string

Parameter	Type	Required / Optional	Description
options.certId	string	required	Certificate ID
options.algorithm	string	required	Hash algorithm
options.rootTag	string	required	Root tag of XML section to sign
options.insertionTag	string	optional	Tag where to insert the signature

certificate.HashAlg

Note: The content in this help topic pertains to SuiteScript 2.0.

Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	The hash algorithm. Supported digest methods are SHA1, SHA256, SHA384, and SHA512 for RSA and ECDSA encryption algorithms and SHA1 and SHA256 for DSA.
Type	enum
Module	N/crypto/certificate Module
Sibling Module Members	N/crypto/certificate Module Members
Since	2019.1

Values

- SHA1
- SHA256
- SHA384
- SHA512

N/currency Module

Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/currency module when you want to work with exchange rates within your NetSuite account. You can use this module to find the exchange rate between two currencies based on a certain date.

To use multiple currencies, the Multiple Currencies feature must be enabled. For information on enabling this feature, see the help topic [Enabling the Multiple Currencies Feature](#).

Note: Currency formatting is handled by the [N/format Module](#).

- [N/currency Module Member](#)

- N/currency Module Script Samples

N/currency Module Member

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	currency.exchangeRate(options)	number	Client and server-side scripts	Returns an exchange rate between two currencies.

N/currency Module Script Samples

The following script samples demonstrate how to use the features of the N/currency module.

Sample 1: Obtain an exchange rate

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to obtain the exchange rate between the Canadian dollar and the U.S. dollar on a specific date.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/currency'], function(currency) {
6     function getUSDFromCAD() {
7         var canadianAmount = 100;
8         var rate = currency.exchangeRate({
9             source: 'CAD',
10            target: 'USD',
11           date: new Date('7/28/2015')
12        });
13
14        var usdAmount = canadianAmount * rate;
15    }
16
17    getUSDFromCAD();
18 });

```

currency.exchangeRate(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Method used to return the exchange rate between two currencies based on a certain date. The source currency is looked up relative to the target currency on the effective date. For example, if use British pounds for the source and US dollars for the target and the method returns '1.52', this means that if you were to enter an invoice today for a GBP customer in your USD subsidiary, the rate would be 1.52.</p> <p>The exchange rate values are sourced from the Currency Exchange Rate record.</p>
---------------------------	--

	Note: The Currency Exchange Rate record itself is not a scriptable record.
	Only base currencies can be converted. The target must be a base currency in your NetSuite account. If not, inaccurate values are returned. For details, see the help topic Setting a Base Currency .
Returns	The exchange rate as a decimal number
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/currency Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.				
Parameter	Type	Required / Optional	Description	Since
options.date	Date	optional	<ul style="list-style-type: none"> ■ Pass in a new Date object. For example, date: new Date('7/28/2015') ■ If date is not specified, then it defaults to today (the current date). ■ The date determines the exchange rate in effect. If there are multiple rates, it is the latest entry on that date. ■ Use the same date format as your NetSuite account. 	2015.2
options.source	number string	required	<ul style="list-style-type: none"> ■ The internal ID or three-letter ISO code for the currency you are converting from. ■ For example, you can use either 1 (internal ID) or USD (currency code). ■ If the Multiple Currencies feature is enabled, from your account, you can view a list of all the currency internal IDs and ISO codes at Lists > Accounting > Currencies. 	2015.2
options.target	number string	required	<ul style="list-style-type: none"> ■ The internal ID or three-letter ISO code for the currency you are converting to. 	2015.2

Errors

Error Code	Message	Thrown If
MISSING_REQD_ARGUMENT	exchangeRate: Missing a required argument: <source/target>	The source or target argument is missing.
SSS_INVALID_CURRENCY_ID	You have entered an invalid currency symbol or internal ID: <target/source>	The source or target argument is invalid.

Error Code	Message	Thrown If
		If the Multiple Currencies feature is enabled, from your account, you can view a list of currency internal IDs and ISO codes at Lists > Accounting > Currencies.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currency Module Script Sample.

```

1 //Add additional code
2 ...
3 var canadianAmount = 100;
4 var rate = currency.exchangeRate({
5   source: 'CAD',
6   target: 'USD',
7   date: new Date('7/28/2015')
8 });
9 var usdAmount = canadianAmount * rate;
10 ...
11 //Add additional code

```

N/currentRecord Module

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

You use the N/currentRecord module to access the record that is active in the current client-side context. This module is always a dynamic object and mode of work is always dynamic, not deferred dynamic/standard. For more information, see the help topic [SuiteScript 2.0 – Standard and Dynamic Modes](#). Be aware that when the current record is in view mode it cannot be edited; it is a read-only record when in view mode. As such, any set APIs do not work on the current record in view mode.

You can use the currentRecord module in the following types of scripts:

- **Entry point client scripts** — These scripts use the @NScriptType ClientScript annotation. (For details, see the help topic [SuiteScript 2.0 JSDoc Validation](#).) The system automatically provides this type of script with a `currentRecord.CurrentRecord` object that represents the current record. For this reason, an entry point client script does not have to explicitly load the currentRecord module. To access the currentRecord object, create a variable and initialize it to the value of the `scriptContext.currentRecord` property, which is available in each of the [SuiteScript 2.0 Client Script Entry Points and API](#). For an example, see the help topic [SuiteScript Client Script Sample](#).
- **Client-side custom modules** — These scripts do not use an @NScriptType annotation (see the help topic [SuiteScript 2.0 Custom Modules](#)). For these scripts, you must manually load the currentRecord module by naming it in the script's define statement. Additionally, you must actively retrieve a `currentRecord.CurrentRecord` object by using the `currentRecord.get()` or `currentRecord.get.promise()` method. For an example, see N/currentRecord Module Script Samples.

Like the [N/record Module](#), the currentRecord module provides access to body and sublist fields. However, the record module is recommended for server scripts and for cases where a client-side script needs to interact with a record other than the currently active record. By contrast, the currentRecord module is recommended for client-side scripts that need to interact with the currently active record.

Additionally, the functionality of the two modules varies slightly. For example, the currentRecord module does not permit the editing of subrecords, although subrecords can be retrieved in view mode. For additional details, see the following topics:

- N/currentRecord Module Members
- Column Object Members
- CurrentRecord Object Members
- Field Object Members
- Sublist Object Members
- N/currentRecord Module Script Sample

i Note: SuiteScript supports working with standard NetSuite records and with instances of custom record types. Supported standard record types are described in the [SuiteScript Records Browser](#). Refer also to [SuiteScript Supported Records](#). For help interacting with an instance of a custom record type, see the help topic [Custom Record](#).

N/currentRecord Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	currentRecord.Column	Object	Client scripts	Encapsulates a column of a sublist on the current record.
	currentRecord.CurrentRecord	Object	Client scripts	Represents the record active on the current page.
	currentRecord.Field	Object	Client scripts	Represents a body or sublist field.
	currentRecord.Sublist	Object	Client scripts	Represents a sublist.
Method	currentRecord.get()	currentRecord.CurrentRecord	Client scripts	Retrieves a record object that represents the current record.
	currentRecord.get.promise()	Promise	Client scripts	Retrieves a promise for an object that represents the current record.

Column Object Members

The following members are called on the `currentRecord.Column` object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	Column.id	string (read-only)	Client scripts	Returns the internal ID of the column.
	Column.isDisabled	boolean true false	Client scripts	Indicates whether the column is disabled.
	Column.isMandatory	boolean true false	Client scripts	Indicates whether the column is mandatory.
	Column.label	string (read-only)	Client scripts	Returns the UI label for the column.
	Column.sublistId	string (read-only)	Client scripts	Returns the internal ID of the standard or custom sublist that contains the column.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Column.type	string (read-only)	Client scripts	Returns the column type.

CurrentRecord Object Members

The following members are called on the `currentRecord.CurrentRecord` object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	CurrentRecord.cancelLine(options)	currentRecord.CurrentRecord	Client scripts	Cancels the changes made to the currently selected line.
	CurrentRecord.commitLine(options)	currentRecord.CurrentRecord	Client scripts	Commits the currently selected line.
	CurrentRecord.findMatrixSublistLineWithValue(options)	number	Client scripts	Returns the line number of the first line that contains the specified value in the matrix column.
	CurrentRecord.findSublistLineWithValue(options)	number	Client scripts	Gets the line number for the first occurrence of a field value in a sublist.
	CurrentRecord.getCurrentMatrixSublistValue(options)	number Date string array boolean true false	Client scripts	Gets the value for the currently selected line in the matrix.
	CurrentRecord.getCurrentSublistIndex(options)	number	Client scripts	Gets the line number of the currently selected line.
	CurrentRecord.getCurrentSublistSubrecord(options)	currentRecord.CurrentRecord	Client scripts	Gets the subrecord for the associated sublist field on the current line. The subrecord object is retrieved in view mode.
	CurrentRecord.getCurrentSublistText(options)	number Date string array boolean true false	Client scripts	Gets the value of the field in the currently selected line by text representation.
	CurrentRecord.getCurrentSublistValue(options)	number Date string array boolean true false	Client scripts	Gets the value of the field in the currently selected line.
	CurrentRecord.getField(options)	currentRecord.Field	Client scripts	Gets a field object from the record.
	CurrentRecord.getLineCount(options)	number	Client scripts	Returns the number of lines in the sublist.
	CurrentRecord.getMatrixHeaderCount(options)	number	Client scripts	Returns the number of columns for the specified matrix.
	CurrentRecord.getMatrixHeaderField(options)	currentRecord.Field	Client scripts	Gets the field for the specified header in the matrix.
	CurrentRecord.getMatrixHeaderValue(options)	number Date string array boolean true false	Client scripts	Gets the value for the associated header in the matrix.
	CurrentRecord.getMatrixSublistField(options)	currentRecord.Field	Client scripts	Gets the field for the specified sublist in the matrix.
	CurrentRecord.getMatrixSublistValue(options)	number Date string array boolean true false	Client scripts	Gets the value for the associated field in the matrix.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	CurrentRecord.getSublist(options)	currentRecord.Sublist	Client scripts	Gets the specified sublist object.
	CurrentRecord.getSublistField(options)	currentRecord.Field	Client scripts	Gets the specified field object from the sublist.
	CurrentRecord.getSublistText(options)	string	Client scripts	Gets the value of the field in a sublist by a string representation.
	CurrentRecord.getSublistValue(options)	number Date string array boolean true false	Client scripts	Gets the value of the field in a sublist.
	CurrentRecord.getSubrecord(options)	currentRecord.CurrentRecord	Client scripts	Gets the subrecord associated with the field. The subrecord object is retrieved in view mode.
	CurrentRecord.getText(options)	string	Client scripts	Gets the value of the field by a string representation.
	CurrentRecord.getValue(options)	number Date string array boolean true false	Client scripts	Gets the value of the field.
	CurrentRecord.hasCurrentSublistSubrecord(options)	boolean true false	Client scripts	Returns a value indicating whether the associated sublist field has a subrecord on the current line.
	CurrentRecord.hasSublistSubrecord(options)	boolean true false	Client scripts	Returns a value indicating whether the associated sublist field contains a subrecord.
	CurrentRecord.hasSubrecord(options)	boolean true false	Client scripts	Indicates whether the field has a subrecord.
	CurrentRecord.insertLine(options)	currentRecord.CurrentRecord	Client scripts	Inserts a new line in a sublist.
	CurrentRecord.removeCurrentSublistSubrecord(options)	currentRecord.CurrentRecord	Client scripts	Removes the subrecord for the associated sublist field on the current line.
	CurrentRecord.removeLine(options)	currentRecord.CurrentRecord	Client scripts	Removes a line from a sublist.
	CurrentRecord.removeSubrecord(options)	currentRecord.CurrentRecord	Client scripts	Removes the subrecord associated with the field.
	CurrentRecord.selectLine(options)	void	Client scripts	Selects a line item in a sublist.
	CurrentRecord.selectNewLine(options)	currentRecord.CurrentRecord	Client scripts	Selects a new line at the end of the sublist.
	CurrentRecord.setCurrentMatrixSublistValue(options)	currentRecord.CurrentRecord	Client scripts	Sets the value for the currently selected line in the matrix.
	CurrentRecord.setCurrentSublistText(options)	currentRecord.CurrentRecord	Client scripts	Sets the value of the field in the currently selected line using a string representation.
	CurrentRecord.setCurrentSublistValue(options)	currentRecord.CurrentRecord	Client scripts	Sets the value of the field in the currently selected line.
	CurrentRecord.setMatrixHeaderValue(options)	currentRecord.CurrentRecord	Client scripts	Sets the value for the associated header in the matrix.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	CurrentRecord.setMatrixSublistValue(options)	currentRecord. CurrentRecord	Client scripts	Sets the value for the associated field in the matrix.
	CurrentRecord.setText(options)	currentRecord. CurrentRecord	Client scripts	Sets the value of the field using a string representation.
	CurrentRecord.setValue(options)	currentRecord. CurrentRecord	Client scripts	Sets the value of the field.
Property	CurrentRecord.id	number (read-only)	Client scripts	Returns the internal record ID.
	CurrentRecord.isDynamic	boolean true false (read-only)	Client scripts	Indicates whether the record is dynamic.
	CurrentRecord.type	string (read-only)	Client scripts	Returns the record type.

Field Object Members

The following members are called on the `currentRecord.Field` object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Field.getSelectOptions(options)	array	Client scripts	Returns an array of available options on a standard or custom select, multiselect, or radio field as key-value pairs. Only the first 1,000 available options are returned.
	Field.insertSelectOption(options)	void	Client scripts	Inserts an option into certain types of select and multiselect fields. This method is usable only in fields that were added by a front-end Suitelet or beforeLoad user event script.
	Field.removeSelectOption(options)	void	Client scripts	Removes an option from certain types of select and multiselect fields. This method is usable only in fields that were added by a front-end Suitelet or beforeLoad user event script. It is supported only in client scripts..
Object	Field.id	string (read-only)	Client scripts	Returns the internal ID of a standard or custom body or sublist field.
	Field.isDisabled	boolean true false	Client scripts	Returns true if the standard or custom field is disabled on the record form, or false otherwise.
	Field.isDisplay	boolean true false	Client scripts	Returns true if the field is set to display on the record form, or false otherwise. This property is read-only for sublist fields.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Field.isMandatory	boolean true false	Client scripts	Returns true if the standard or custom field is mandatory on the record form, or false otherwise.
	Field.isPopup	boolean true false (read-only)	Client scripts	Returns true if the field is a popup list field, or false otherwise.
	Field.isReadOnly	boolean true false	Client scripts	Returns true if the field on the record form cannot be edited, or false otherwise. For textarea fields, this property can be read or written to. For all other fields, this property is read-only.
	Field.isVisible	boolean true false (read-only)	Client scripts	Returns true if the field is visible on the record form, or false otherwise.
	Field.label	string (read-only)	Client scripts	Returns the UI label for a standard or custom field body or sublist field.
	Field.sublistId	string (read-only)	Client scripts	Returns the ID of the sublist associated with the specified sublist field.
	Field.type	string (read-only)	Client scripts	Returns the type of a body or sublist field.

Sublist Object Members

The following members are called on the [currentRecord.Sublist](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Sublist.getColumn (options)	currentRecord.Column	Client scripts	Returns a column in the sublist.
Property	Sublist.id	string (read-only)	Client scripts	Returns the internal ID of the sublist.
	Sublist.isChanged	boolean true false (read-only)	Client scripts	Indicates whether the sublist has changed on the current record form.
	Sublist.isDisplay	boolean true false (read-only)	Client scripts	Indicates whether the sublist is displayed on the current record form.
	Sublist.type	string (read-only)	Client scripts	Returns the sublist type.

N/currentRecord Module Script Sample

i Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample is a custom module client script named clientDemo.js. This script updates fields on the current record. After you upload clientDemo.js to a NetSuite account, it can be called by other scripts, as shown in the subsequent sample.

Because clientDemo.js is a custom module script, it must manually load the currentRecord module by naming it in the define statement. Additionally, it must actively retrieve a CurrentRecord object. It does so by using the currentRecord.get() method.

```

1 /**
2  * @NApiVersion 2.0
3 */
4
5 define(['N/currentRecord'], function(currentRecord) {
6     return({
7         test_set_getValue: function() {
8             var record = currentRecord.get();
9             record.setValue({
10                 fieldId: 'custpage_textfield',
11                 value: 'Body value',
12                 ignoreFieldChange: true,
13                 forceSyncSourcing: true
14             });
15             var actValue = record.getValue({
16                 fieldId: 'custpage_textfield'
17             });
18             record.setValue({
19                 fieldId: 'custpage_resultfield',
20                 value: actValue,
21                 ignoreFieldChange: true,
22                 forceSyncSourcing: true
23             });
24         },
25
26         test_set_getCurrentSublistValue: function() {
27             var record = currentRecord.get();
28             record.setCurrentSublistValue({
29                 sublistId: 'sitecategory',
30                 fieldId: 'custpage_subtextfield',
31                 value: 'Sublist Value',
32                 ignoreFieldChange: true,
33                 forceSyncSourcing: true
34             });
35             var actValue = record.getCurrentSublistValue({
36                 sublistId: 'sitecategory',
37                 fieldId: 'custpage_subtextfield'
38             });
39             record.setValue({
40                 fieldId: 'custpage_sublist_resultfield',
41                 value: actValue,
42                 ignoreFieldChange: true,
43                 forceSyncSourcing: true
44             });
45         },
46     });
47 });

```



Note: This script sample uses the `define` function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the `require` function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample is a user event script deployed on a non-inventory item record. Before the record loads, the script updates the form used by the record to add new text fields, a sublist, and buttons that call the `clientDemo.js` methods. The buttons access the current record and set values for some of the form's fields. The use case for this sample is to set up a page, adding fields and buttons, so that you can use the code you made in the first sample, and see the fields and buttons in action.

```

1  /**
2   * @NApiVersion 2.0
3   * @NScriptType UserEventScript
4   * @NModuleScope SameAccount
5   */
6
7 define([], function() {
8     return {
9         beforeLoad: function (params)
10        {
11            var form = params.form;
12
13            var textfield = form.addField({
14                id: 'custpage_textfield',
15                type: 'text',
16                label: 'Text'
17            });
18            var resultfield = form.addField({
19                id: 'custpage_resultfield',
20                type: 'text',
21                label: 'Result'
22            });
23            var sublistResultfield = form.addField({
24                id: 'custpage_sublist_resultfield',
25                type: 'text',
26                label: 'Sublist Result Field'
27            });
28
29            var sublistObj = form.getSublist({
30                id: 'sitecategory'
31            });
32            var subtextfield = sublistObj.addField({
33                id: 'custpage_subtextfield',
34                type: 'text',
35                label: 'Sublist Text Field'
36            });
37
38            form.clientScriptModulePath = './clientDemo.js';
39            form.addButton({
40                id: 'custpage_custombutton',
41                label: 'SET_GET_VALUE',
42                functionName: 'test_set_getValue'
43            });
44            form.addButton({
45                id: 'custpage_custombutton2',
46                label: 'SET_GETCURRENTSUBLISTVALUE',
47                functionName: 'test_set_getCurrentSublistValue'
48            });
49        }
50    };
51 });

```



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following script sample shows how to use the forceSyncSourcing parameter.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/currentRecord'], function(currentRecord){
6     var rec = currentRecord.get();
7     rec.selectNewLine({
8         sublistId: 'item'
9     });
10    rec.setCurrentSublistValue({
11        sublistId: 'item',
12        fieldId: 'item',
13        value: 39,
14        forceSyncSourcing:true
15    });
16    rec.setCurrentSublistValue({
17        sublistId: 'item',
18        fieldId: 'quantity',
19        value: 1,
20        forceSyncSourcing:true
21    });
22    rec.commitLine({sublistId: 'item'});
23 })

```

currentRecord.Column



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a column of a sublist on the current record. For a complete list of this object's properties, see Column Object Members .
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var objColumn = objSublist.getColumn({
4     fieldId: 'item'
5 });
6 ...
7 //Add additional code

```

Column.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of the column.
Type	string (read-only)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code ...
2 var columnid = objColumn.id;
3 ...
4 //Add additional code

```

Column.isDisabled



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the column is disabled.
Type	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/currentRecord Module
Sibling Object Members	Column Object Members
Since	2020.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistObj = recordObj.getSublist({
4     sublistId: 'sublistId'
5 });
6 var columnObj = sublistObj.getColumn({

```

```

7     fieldId: 'columnId'
8 });
9 //set
10 columnObj.isDisabled = true;
11 //get
12 var isDisabled = columnObj.isDisabled;
13 })...
14 //Add additional code.

```

Column.isMandatory



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the column is mandatory.
Type	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/currentRecord Module
Sibling Object Members	Column Object Members
Since	2020.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistObj = recordObj.getSublist({
4     sublistId: 'sublistId'
5 });
6 var columnObj = sublistObj.getColumn({
7     fieldId: 'columnId'
8 });
9 //set
10 columnObj.isMandatory = true;
11 //get
12 var isMandatory = columnObj.isMandatory;
13 })...
14 //Add additional code.

```

Column.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of the column.
Type	string (read-only)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .

Module	N/currentRecord Module
Since	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var columnlabel = objColumn.label;
4 ...
5 //Add additional code

```

Column.sublistId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of the standard or custom sublist that contains the column.
Type	string (read-only)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var sublistid = objColumn.sublistId;
4 ...
5 //Add additional code

```

Column.type

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the column type.
Type	string (read-only)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module

Since	2016.2
--------------	--------

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var columntype = objColumn.type;
4 ...
5 //Add additional code

```

currentRecord.CurrentRecord

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the record active on the current page.
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Syntax

Important: The following code snippets show the syntax for this member. These snippets are not a functional examples. For a complete script example, see [N/currentRecord Module Script Sample](#) and [SuiteScript Client Script Sample](#).

The following snippet shows the retrieval of a currentRecord object in a custom module where the currentRecord was explicitly loaded.

```

1 //Add additional code
2 ...
3 var objRecord = currentRecord.get();
4 ...
5 //Add additional code

```

In an entry point client script, you do not have use the get method to retrieve the current record. (An entry point client script is one that uses the @NScriptType ClientScript annotation.) In these scripts, a currentRecord object is automatically created when the script is loaded. It is part of the context object that passed to each of the client script type's entry points. However, you do have to create a variable to represent the current record, as shown in the following snippet.

```

1 //Add additional code
2 ...
3
4 function pageInit(context) {
5     var currentRec = context.currentRecord;
6 ...
7 //Add additional code

```

CurrentRecord.cancelLine(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Cancels the currently selected line on a sublist.
Returns	The currentRecord.CurrentRecord object that called the method.
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.cancelLine({
4     sublistId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.commitLine(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Commits the currently selected line on a sublist.
---------------------------	---

Returns	The <code>currentRecord.CurrentRecord</code> object that called the method.
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
<code>options.ignoreRecalc</code>	boolean true false	optional	If set to true, scripting recalculation is ignored.	2016.2

Errors

Error Code	Thrown If
<code>SSS_MISSING_REQD_ARGUMENT</code>	A required argument is missing or undefined.
<code>SSS_INVALID_SUBLIST_OPERATION</code>	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
--

```

1 //Add additional code
2 ...
3 objRecord.commitLine({
4   sublistId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.findMatrixSublistLineWithValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Returns the line number of the first instance where a specified value is found in a specified column of the matrix.
Returns	number

Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2
options.value	number	required	The value to search for.	2016.2
options.column	number	required	The column number of the field.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
--

```

1 //Add additional code
2 ...
3 var lineNumber = objRecord.findMatrixSublistLineWithValue({
4     sublistId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.findSublistLineWithValue(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the line number for the first occurrence of a field value in a sublist.
---------------------------	---

Returns	A line number as a number, or -1 if not found.
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2
options.value	number Date string array boolean true false	optional	The value to search for.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or not defined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var lineNumber = objRecord.findSublistLineWithValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   value: 233
7 });
8 ...
9 //Add additional code

```

CurrentRecord.getCurrentMatrixSublistValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the value for the currently selected line in the matrix. Gets a numeric value for rate and ratehighprecision fields.
Returns	number Date string array boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2
options.column	number	required	The column number for the matrix field.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var matrixValue = objRecord.getCurrentMatrixSublistValue({

```

```

4     sublistId: 'item',
5     fieldId: 'item',
6     column: 12
7 });
8 ...
9 //Add additional code

```

CurrentRecord.getCurrentSublistIndex(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the line number of the currently selected line.
Returns	number
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var currIndex = objRecord.getCurrentSublistIndex({
4   sublistId: 'item'
5 });

```

```

6 ...
7 //Add additional code

```

CurrentRecord.getCurrentSublistSubrecord(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the subrecord for the associated sublist field on the current line.
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object. If no subrecord instance exists, the system creates one. For more information, see the help topic [Subrecord Scripting in SuiteScript 2.0 Compared With 1.0](#).

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
<code>options.fieldId</code>	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var objSubrecord = objRecord.getCurrentSublistSubrecord({
4   sublistId: 'item',
5   fieldId: 'item'
6 });
7 ...
8 //Add additional code

```

CurrentRecord.getCurrentSublistText(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a text representation of the field value in the currently selected line.
Returns	string Note: For multiselect fields, returns an array.
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fieldName = objRecord.getCurrentSublistText({
4     sublistId: 'item',

```

```

5   fieldId: 'item'
6 });
7 ...
8 //Add additional code

```

CurrentRecord.getCurrentSublistValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a sublist field on the currently selected sublist line.
Returns	number Date string array boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code

```

```

2 ...
3 var sublistValue = objRecord.getCurrentSublistValue({
4   sublistId: 'item',
5   fieldId: 'item'
6 });
7 ...
8 //Add additional code

```

CurrentRecord.getField(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a field object from a record.
Returns	currentRecord.Field
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See the help topic How do I find a field's internal ID?	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var objField = objRecord.getField({
4   fieldId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.getLineCount(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the number of lines in a sublist.
Returns	number
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var numLines = objRecord.getLineCount({
4     sublistId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.getMatrixHeaderCount(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the number of columns for the specified matrix.
Returns	number
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None

Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
--

```

1 //Add additional code
2 ...
3 var numLines = objRecord.getMatrixHeaderCount({
4   sublistId: 'item',
5   fieldId: 'item'
6 });
7 ...
8 //Add additional code

```

CurrentRecord.getMatrixHeaderField(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Gets the field for the specified header in the matrix.
Returns	currentRecord.Field
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module

Since	2016.2
--------------	--------

Parameters

i	Note: The options parameter is a JavaScript object.
----------	--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2
options.column	number	required	The column number for the field.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

!	Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
----------	--

```

1 //Add additional code
2 ...
3 var objField = objRecord.getMatrixHeaderField({
4   sublistId: 'item',
5   fieldId: 'item',
6   column: 12
7 });
8 ...
9 //Add additional code

```

CurrentRecord.getMatrixHeaderValue(options)

i	Note: The content in this help topic pertains to SuiteScript 2.0.
----------	--

Method Description	Gets the value for the associated header in the matrix.
Returns	number Date string array boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None

Module	N/currentRecord Module
Since	2016.2

Parameters

i	Note: The options parameter is a JavaScript object.
----------	--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2
options.column	number	required	The column number for the field.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [currentRecord Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var value = objRecord.getMatrixHeaderValue({
4     sublistId: 'item',
5     fieldId: 'item',
6     column: 12
7 });
8 ...
9 //Add additional code

```

CurrentRecord.getMatrixSublistField(options)

i	Note: The content in this help topic pertains to SuiteScript 2.0.
----------	--

Method Description	Gets the field for the specified sublist in the matrix.
Returns	currentRecord.Field
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .

Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2
options.column	number	required	The column number for the field.	2016.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
--

```

1 //Add additional code
2 ...
3 var objField = objRecord.getMatrixSublistField({
4   sublistId: 'item',
5   fieldId: 'item',
6   column: 12,
7   line: 3
8 });
9 ...
10 //Add additional code

```

CurrentRecord.getMatrixSublistValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Gets the value for the associated field in the matrix.
---------------------------	--

Returns	number Date string array boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2
options.column	number	required	The column number for the field.	2016.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
--

```

1 //Add additional code
2 ...
3 var value = objRecord.getMatrixSublistValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   column: 12,
7   line: 3
8 });
9 ...
10 //Add additional code

```

CurrentRecord.getSublist(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the specified sublist.
Returns	<code>currentRecord.Sublist</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var objSublist = objRecord.getSublist({
4     sublistId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.getSublistField(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a field object from a sublist. Use Record.getCurrentSublistField(options) if you are working with the current and non-committed line.
Returns	<code>currentRecord.Field</code>
Supported Script Types	Client scripts

	For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

i	Note: The options parameter is a JavaScript object.
----------	--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
--

```

1 //Add additional code
2 ...
3 var objField = objRecord.getSublistField({
4   sublistId: 'item',
5   fieldId: 'item',
6   line: 3
7 });
8 ...
9 //Add additional code

```

CurrentRecord.getSublistText(options)

i	Note: The content in this help topic pertains to SuiteScript 2.0.
----------	--

Method Description	Returns the value of a sublist field in a text representation.
---------------------------	--

	Gets a string value with a "%" for rate and ratehighprecision fields.
Returns	string
	<p> Note: For multiselect fields, returns an array.</p>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
---	--

```

1 //Add additional code
2 ...
3 var sublistFieldName = objRecord.getSublistText({
4   sublistId: 'item',
5   fieldId: 'item',
6   line: 3
7 });
8 ...
9 //Add additional code

```

CurrentRecord.getSublistValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a sublist field. Gets a numeric value for rate and ratehighprecision fields.
Returns	number Date string array boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var quantity = record.getSublistValue({
4     sublistId: 'item',

```

```

5   fieldId: 'quantity',
6   line: 0
7 });
8 ...
9 //Add additional code

```

CurrentRecord.getSubrecord(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the subrecord associated with the field.
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See the help topic How do I find a field's internal ID?	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
FIELD_1_IS_NOT_A_SUBRECORD_FIELD	The specified field is not a subrecord field.
FIELD_1_IS_DISABLED_YOU_CANNOT_APPLY_SUBRECORD_OPERATION_ON_THIS_FIELD	The specified field is disabled.
SSS_INVALID_FIELD_ON_SUBRECORD_OPERATION	The specified fieldId does not refer to a subrecord.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code

```

```

2 ...
3 var sublistFieldValue = objRecord.getSubrecord({
4   fieldId: 'subrecord'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.getText(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the text representation of a field value. Gets a string value with a "%" for rate and ratehighprecision fields.
Returns	string  Note: For multiselect fields, returns an array.
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See the help topic How do I find a field's internal ID?	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```
1 //Add additional code
```

```

2 ...
3 var fieldidname = objRecord.getText({
4   fieldId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.getValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a field.
Returns	number Date string array boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See the help topic How do I find a field's internal ID?	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var value = objRecord.getValue({
4   fieldId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.hasCurrentSublistSubrecord(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a value indicating whether the associated sublist field has a subrecord on the current line. This method can only be used on dynamic records.
Returns	boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a subrecord. See the help topic How do I find a field's internal ID?	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var hasSubrecord = objRecord.hasCurrentSublistSubrecord({
4   sublistId: 'item',
5   fieldId: 'item'
6 });
7 ...
8 //Add additional code

```

CurrentRecord.hasSublistSubrecord(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a value indicating whether the associated sublist field contains a subrecord.
---------------------------	---

Returns	boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a subrecord. See the help topic How do I find a field's internal ID?	2016.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2

Syntax

! Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .

```

1 //Add additional code
2 ...
3 var hasSubrecord = objRecord.hasSublistSubrecord({
4   sublistId: 'item',
5   fieldId: 'item',
6   line: 3
7 });
8 ...
9 //Add additional code

```

CurrentRecord.hasSubrecord(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a value indicating whether the field contains a subrecord.
Returns	boolean true false
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .

Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of the field that may contain a subrecord. See the help topic How do I find a field's internal ID?	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var hasSubrecord = objRecord.hasSubrecord({
4     fieldId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.insertLine(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Inserts a sublist line.
Returns	currentRecord.CurrentRecord
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist.	2016.2

Parameter	Type	Required / Optional	Description	Since
			This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	
options.line	number	required	The line number to insert. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2
options.ignoreRecalc	boolean true false	optional	If set to true, scripting recalculation is ignored. The default value is false.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.insertLine({
4     sublistId: 'item',
5     line: 3,
6     ignoreRecalc: true
7 });
8 ...
9 //Add additional code

```

CurrentRecord.removeCurrentSublistSubrecord(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the subrecord for the associated sublist field.
Returns	currentRecord.CurrentRecord
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	<p>The internal ID of the sublist.</p> <p>This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser.</p>	2016.2
options.fieldId	string	required	<p>The internal ID of a standard or custom sublist field.</p> <p>See the help topic How do I find a field's internal ID?</p>	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.removeCurrentSublistSubrecord({
4   sublistId: 'item',
5   fieldId: 'item'
6 });
7 ...
8 //Add additional code

```

CurrentRecord.removeLine(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes a sublist line.
Returns	currentRecord.CurrentRecord
Supported Script Types	<p>Client scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Client Script Type.</p>
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist.	2016.2

Parameter	Type	Required / Optional	Description	Since
			This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	
options.line	number	required	The line number of the sublist to remove. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2
options.ignoreRecalc	boolean true false	optional	If set to true, scripting recalculation is ignored. The default value is false.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.removeLine({
4   sublistId: 'item',
5   line: 3,
6   ignoreRecalc: true
7 });
8 ...
9 //Add additional code

```

CurrentRecord.removeSubrecord(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the subrecord for the associated field.
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See the help topic How do I find a field's internal ID?	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.removeSubrecord({
4   fieldid: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.selectLine(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Selects an existing line in a sublist.
Returns	currentRecord.CurrentRecord
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2

Parameter	Type	Required / Optional	Description	Since
options.line	number	required	The line number to select in the sublist. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var record = CurrentRecord.selectLine({
4     sublistId: 'item',
5     line: 3
6 });
7 ...
8 //Add additional code

```

CurrentRecord.selectNewLine(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Selects a new line at the end of a sublist.
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist.	2016.2

Parameter	Type	Required / Optional	Description	Since
			This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.selectNewLine({
4   sublistId: 'item'
5 });
6 ...
7 //Add additional code

```

CurrentRecord.setCurrentMatrixSublistValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the line currently selected in the matrix. Sets a numeric value for rate and ratehighprecision fields. This method is not available for standard records.
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist that contains the matrix.	2016.2

Parameter	Type	Required / Optional	Description	Since
			This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	
options.fieldId	string	required	The internal ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2
options.column	number	required	The column number for the field.	2016.2
options.value	number Date string array boolean true false	required	<p>The value to set the field to.</p> <p>The value type must correspond to the field type being set. For example:</p> <ul style="list-style-type: none"> ■ Text, Radio and Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2016.2
options.ignoreFieldChange	boolean true false	optional	<p>If set to true, the field change and the secondary event is ignored.</p> <p>By default, this value is false.</p>	2016.2
options.forceSyncSourcing	boolean true false	optional	<p>Indicates whether to perform field sourcing synchronously.</p> <p>If set to true, sources dependent field information for empty fields synchronously.</p> <p>Defaults to false – dependent field values are not sourced synchronously.</p>	2019.1

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.setCurrentMatrixSublistValue({
```

```

4 |     sublistId: 'item',
5 |     fieldId: 'item',
6 |     column: 3,
7 |     value: false,
8 |     ignoreFieldChange: true,
9 |     forceSyncSourcing: true
10 |   });
11 | ...
12 | //Add additional code

```

CurrentRecord.setCurrentSublistText(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the field in the currently selected line by a text representation. Sets a string value with a "%" for rate and ratehighprecision fields.
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2
options.text	string	required	The text to set the value to.	2016.2
options.ignoreFieldChange	boolean true false	optional	If set to true, the field change and the secondary event is ignored. By default, this value is false.	2016.2
options.forceSyncSourcing	boolean true false	optional	Indicates whether to perform field sourcing synchronously.	2019.1

Parameter	Type	Required / Optional	Description	Since
			If set to true, sources dependent field information for empty fields synchronously. Defaults to false – dependent field values are not sourced synchronously.	

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
A_SCRIPT_IS_ATTEMPTING_TO_EDIT_THE_1_SUBLIST_THIS_SUBLIST_IS_CURRENTLY_IN_READONLY_MODE_AND_CANNOT_BE_EDITED_CALL_YOUR_NETSUITE_ADMINISTRATOR_TO_DISABLE_THIS_SCRIPT_IF_YOU_NEED_TO_SUBMIT_THIS_RECORD	A user tries to edit a read-only sublist field.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.setCurrentSublistText({
4   sublistId: 'item',
5   fieldId: 'item',
6   text: 'value',
7   ignoreFieldChange: true,
8   forceSyncSourcing: true
9 });
10 ...
11 //Add additional code

```

CurrentRecord.setCurrentSublistValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the field in the currently selected line.
Important: When you edit a sublist line with SuiteScript, it triggers an internal validation of the sublist line. If the line validation fails, the script also fails. For example, if your script edits a closed catch up period, the validation fails and prevents SuiteScript from editing the closed catch up period.	
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .

Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID?	2016.2
options.value	number Date string array boolean true false	required	The value to set the field to. The value type must correspond to the field type being set. For example: <ul style="list-style-type: none"> ■ Text, Radio and Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2016.2
options.ignoreFieldChange	boolean true false	optional	If set to true, the field change and the secondary event is ignored. By default, this value is false.	2016.2
options.forceSyncSourcing	boolean true false	optional	Indicates whether to perform field sourcing synchronously. By default, this value is false.	2019.1

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Error Code	Thrown If
A_SCRIPT_IS_ATTEMPTING_TO_EDIT_THE_1_SUBLIST_THIS_SUBLIST_IS_CURRENTLY_IN_READONLY_MODE_AND_CANNOT_BE_EDITED_CALL_YOUR_NETSUITE_ADMINISTRATOR_TO_DISABLE_THIS_SCRIPT_IF_YOU_NEED_TO_SUBMIT_THIS_RECORD	A user tries to edit a read-only sublist field.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.setCurrentSublistValue({
4     sublistId: 'item',
5     fieldId: 'item',
6     value: true,
7     ignoreFieldChange: true
8 });
9 ...
10 //Add additional code

```

CurrentRecord.setMatrixHeaderValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the associated header in the matrix. Sets a numeric value for rate and ratehighprecision fields.
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
<code>options.fieldId</code>	string	required	The internal ID of the matrix field.	2016.2

Parameter	Type	Required / Optional	Description	Since
			See the help topic How do I find a field's internal ID?	
options.column	number	required	The column number for the field.	2016.2
options.value	number Date string array boolean true false	required	<p>The value to set the field to.</p> <p>The value type must correspond to the field type being set. For example:</p> <ul style="list-style-type: none"> ■ Text, Radio and Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2016.2
options.ignoreFieldChange	boolean true false	optional	<p>If set to true, the field change and the secondary event is ignored.</p> <p>By default, this value is false.</p>	2016.2
options.forceSyncSourcing	boolean true false	optional	<p>Indicates whether to perform field sourcing synchronously.</p> <p>If set to true, sources dependent field information for empty fields synchronously.</p> <p>Defaults to false – dependent field values are not sourced synchronously.</p>	2019.1

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.setMatrixHeaderValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   column: 3,
7   value: false,
8   ignoreFieldChange: true,
9   forceSyncSourcing: true
10 });
11 ...
12 //Add additional code

```

CurrentRecord.setMatrixSublistValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the associated field in the matrix. Sets a numeric value for rate and ratehighprecision fields.
Returns	<code>currentRecord.CurrentRecord</code>
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The internal ID of the matrix field. See the help topic How do I find a field's internal ID?	2016.2
options.column	number	required	The column number for the field.	2016.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2016.2
options.value	number Date string array boolean true false	required	The value to set the field to. The value type must correspond to the field type being set. For example: <ul style="list-style-type: none"> ■ Text, Radio and Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2016.2

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.setMatrixSublistValue({
4     sublistId: 'item',
5     fieldId: 'item',
6     column: 12,
7     line: 3,
8     value: true
9 });
10 ...
11 //Add additional code

```

CurrentRecord.setText(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of the field by a text representation. Sets a string value with a "%" for rate and ratehighprecision fields.
Returns	currentRecord.CurrentRecord
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See the help topic How do I find a field's internal ID?	2016.2
options.text	string	required	The text to change the field value to.	2016.2

Parameter	Type	Required / Optional	Description	Since
options.ignoreFieldChange	boolean true false	optional	If set to true, the field change and the secondary event is ignored. By default, this value is false.	2016.2
options.forceSyncSourcing	boolean true false	optional	Indicates whether to perform field sourcing synchronously. If set to true, sources dependent field information for empty fields synchronously. Defaults to false – dependent field values are not sourced synchronously.	2019.1

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 objRecord.setText({
4   fieldId: 'item',
5   text: 'value',
6   ignoreFieldChange: true,
7   forceSyncSourcing: true
8 });
9 ...
10 //Add additional code

```

CurrentRecord.setValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of a field. Sets a numeric value for rate and ratehighprecision fields.
Returns	currentRecord.CurrentRecord
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	<p>The internal ID of a standard or custom body field.</p> <p>See the help topic How do I find a field's internal ID?</p>	2016.2
options.value	number Date string array boolean true false	required	<p>The value to set the field to.</p> <p>The value type must correspond to the field type being set. For example:</p> <ul style="list-style-type: none"> ■ Text, Radio, Select and Multi-Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2016.2
options.ignoreFieldChange	boolean true false	optional	<p>If set to true, the field change and the secondary event is ignored.</p> <p>By default, this value is false.</p>	2016.2
options.forceSyncSourcing	boolean true false	optional	<p>Indicates whether to perform field sourcing synchronously.</p> <p>If set to true, sources dependent field information for empty fields synchronously.</p> <p>Defaults to false – dependent field values are not sourced synchronously.</p>	2019.1

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
---	--

1 | //Add additional code

```

2 ...
3 objRecord.setValue({
4   fieldId: 'item',
5   value: true,
6   ignoreFieldChange: true,
7   forceSyncSourcing: true
8 });
9 ...
10 //Add additional code

```

CurrentRecord.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of a specific record.
Type	number (read-only)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var recordid = record.id;
4 ...
5 //Add additional code

```

CurrentRecord.isDynamic



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the record is in dynamic mode. For more information, see the help topic SuiteScript 2.0 – Standard and Dynamic Modes . This value is set when the record is created or accessed.
Type	boolean true false (read-only)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 if (record.isDynamic) {
4   ...
5 }
6 ...
7 //Add additional code

```

CurrentRecord.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The current record's type. Note the following: <ul style="list-style-type: none">■ On an instance of a standard record type, this property is represented by a value from the record.Type enum.■ On an instance of a custom record type, this value is populated by the custom record type's string ID. For help finding this ID, see the help topic Custom Record.
Type	string (read-only)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var recordtype = currentRecord.type;
4 ...
5 //Add additional code

```

currentRecord.Field



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a body or sublist field on the current record.
---------------------------	---

	<p>Use the following methods to access the Field object:</p> <ul style="list-style-type: none"> ▪ CurrentRecord.getField(options) ▪ CurrentRecord.getSublistField(options) <p>For a complete list of this object's methods and properties, see N/currentRecord Module.</p>
Supported Script Types	<p>Client scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Client Script Type.</p>
Module	N/currentRecord Module
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var currentRecordField = currentRecord.getField({
4   fieldId: 'entity'
5 });
6 ...
7 //Add additional code
8 }
```

Field.getSelectOptions(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Returns an array of available options on a standard or custom select, multiselect, or radio field as key-value pairs.</p> <p>Important: You can use this method only in dynamic mode. For additional information on dynamic mode, see CurrentRecord.isDynamic.</p>
Returns	<p>array</p> <p>Only the first 1,000 available options are returned in an array.</p> <p>If there are more than 1,000 available options, an empty array [] is returned.</p> <p>This function returns an array in the following format:</p> <pre>1 [{value: 5, text: 'abc'},{value: 6, text: '123'}]</pre> <p>This function returns Type Error if the field is not a supported field for this method.</p>
Governance	None
Supported Script Types	<p>Client scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Client Script Type.</p>
Module	N/currentRecord Module

Since	2016.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.filter	string	Required	The search string to filter the select options that are returned. Note: Filter values are case insensitive.	2016.2
options.operator	string	Required	The following operators are supported: <ul style="list-style-type: none">■ contains (default)■ is■ startswith	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/currentRecord Module Script Sample .
--

```

1 //Add additional code
2 ...
3 var options = objField.getSelectOptions({
4     filter : 'C',
5     operator : 'startswith'
6 });
7 ...
8 //Add additional code

```

Field.insertSelectOption(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Inserts an option into certain types of select and multiselect fields. This method is usable only in select and multiselect fields that were added by a front-end Suitelet or beforeLoad user event script. The IDs for these fields always have a prefix of custpage .
Returns	Void
Governance	None
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.value	string	Required	A string, not visible in the UI, that identifies the option.	2016.2
options.text	string	Required	The label that represents the option in the UI.	2016.2
options.isSelected	boolean	Optional	Determines whether the option is selected by default. If not specified, this value defaults to false.	2016.2

Errors

Error Code	Thrown If
SSS_INVALID_UI_OBJECT_TYPE	A script attempts to use this method on the wrong type of field. This method can be used only on select and multiselect fields whose IDs begin with the prefix custpage .

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example.

```

1 //Add additional code
2 ...
3
4 // Instantiate the field. Note that this method is supported only
5 // on fields whose fieldids have a prefix of custpage.
6
7 var field = call.getField({
8     fieldId: 'custpage_select1field'
9 });
10
11
12 // Insert a new option.
13
14 field.insertSelectOption({
15     value: 'Option1',
16     text: 'alpha'
17 });
18 ...
19 //Add additional code

```

Field.removeSelectOption(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes a select option from certain types of select and multiselect fields. This method is usable only in select fields that were added by a front-end Suitelet or beforeLoad user event script. The IDs for these fields always have a prefix of custpage .
---------------------------	---

Returns	Void
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.value	string	Required	<p>A string, not shown in the UI, that identifies the option.</p> <p>To remove all options from the list, set this field to null, as follows:</p> <pre> 1 ... 2 ... 3 field.removeSelectOption({ 4 value: null, 5 }); 6 ... 7 ... </pre>	2016.2

Errors

Error Code	Thrown If
SSS_INVALID_UI_OBJECT_TYPE	A script attempts to use this method on the wrong type of field. This method can be used only on select and multiselect fields whose IDs begin with the prefix custpage .

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example.
--

```

1 //Add additional code
2 ...
3
4 // Instantiate the field. Note that this method is supported only
5 // on fields whose fieldIds have a prefix of custpage.
6
7 var field = call.getField({
8   fieldId: 'custpage_select1field'
9 });
10
11
12 // Remove the appropriate option.
13
14 field.removeSelectOption({
15   value: 'Option2',
16 });

```

```

17 |
18 | ...
19 | //Add additional code

```

Field.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of a standard or custom body or sublist field.
Type	string (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 | //Add additional code
2 | ...
3 | var id = objField.id;
4 | ...
5 | //Add additional code

```

Field.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the UI label for a standard or custom field body or sublist field.
Type	string (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 | //Add additional code
2 | ...
3 | var label = objField.label;
4 | ...

```

5 | //Add additional code

Field.isMandatory



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns true if the standard or custom field is mandatory on the record form, or false otherwise.
Type	boolean true false
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```
1 //Add additional code
2 ...
3 if (objField.isMandatory) {
4   ...
5 }
6 ...
7 //Add additional code
```

Field.isDisabled



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	This property reflects the display type of a field. A value of true means the field is disabled. A value of false means the field is enabled. Note also: <ul style="list-style-type: none"> ■ If you are working with a body field, you can use this property to change the field's display type. ■ If you are working with a sublist field, you can set this property to true or false, but be aware that this action affects the entire sublist column, even though a sublist field is associated with one line. ■ For both body and sublist fields, you can use Field.isDisabled to determine whether the field is disabled or enabled.
Type	boolean true false
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 if (objField.isDisabled) {
4   ...
5 }
6 ...
7 //Add additional code

```

Field.isPopup



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns true if the field is a popup list field, or false otherwise.
Type	boolean true false (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 if (objField.isPopup) {
4   ...
5 }
6 ...
7 //Add additional code

```

Field.isDisplay



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns true if the field is set to display on the record form, or false otherwise. Fields can be a part of a record even if they are not displayed on the record form. This property is read-only for sublist fields.
Type	boolean true false
Module	N/currentRecord Module
Supported Script Types	Client scripts

	For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 if (objField.isDisplay) {
4   ...
5 }
6 ...
7 //Add additional code

```

Field.isVisible



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns true if the field is visible on the record form, or false otherwise.
Type	boolean true false (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 if (objField.isVisible) {
4   ...
5 }
6 ...
7 //Add additional code

```

Field.isReadOnly



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns true if the field on the record form cannot be edited, or false otherwise. For textarea fields, this property can be read or written to. For all other fields, this property is read-only.
Type	boolean true false

Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 if (objField.isReadOnly) {
4   ...
5 }
6 ...
7 //Add additional code

```

Field.sublistId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the sublist ID for the specified sublist field.
Type	string (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myId = field.sublistId;
4 ...
5 //Add additional code

```

Field.type

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the type of a body or sublist field. For example, the value can return text, date, currency, select, checkbox, and other similar values. For more information on possible return values, see format.Type
-----------------------------	---

	The maximum character limit for select field types is 801.
Type	string (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var type = objField.type;
4 ...
5 //Add additional code

```

currentRecord.Sublist

Object Description	Encapsulates a sublist on the current record. For a complete list of this object's methods and properties, see N/currentRecord Module .
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/currentRecord Module
Since	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var objSublist = currentRecord.getSublist({
4   sublistId: 'item'
5 });
6 ...
7 //Add additional code

```

Sublist.getColumn(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a column in the sublist.
Returns	currentRecord.Column

Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Parameters

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of the column field in the sublist. See the help topic How do I find a field's internal ID?	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [currentRecord Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var objColumn = objSublist.getColumn({
4     fieldId: 'item'
5 });
6 ...
7 //Add additional code

```

Sublist.id

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of the sublist.
Type	string (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```
1 //Add additional code
```

```

2 ...
3 var sublistid = objSublist.id;
4 ...
5 //Add additional code

```

Sublist.isChanged



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the sublist has changed on the current record form.
Type	boolean true false (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 if (objSublist.isChanged) {
4 ...
5 }
6 ...
7 //Add additional code

```

Sublist.isDisplay



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the sublist is displayed on the current record form.
Type	boolean true false (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 if (objSublist.isDisplay) {

```

```

4   ...
5 }
6 ...
7 //Add additional code

```

Sublist.type

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the sublist type.
Type	string (read-only)
Module	N/currentRecord Module
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Since	2016.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var sublisttype = objSublist.type;
4 ...
5 //Add additional code

```

currentRecord.get()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves a currentRecord object that represents the record active on the current page.
Returns	currentRecord.CurrentRecord
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Errors

Error Code	Thrown If
CANNOT_CREATE_RECORD_INSTANCE	The current record page is not scriptable or an error occurred when creating the record object.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/currentRecord Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var record = currentRecord.get();
4 ...
5 //Add additional code

```

currentRecord.get.promise()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves a promise for a currentRecord object that represents the record active on the current page.
	<p>Note: The parameters and errors thrown for this method are the same as those for currentRecord.get(). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/currentRecord Module
Since	2016.2

Errors

Error Code	Thrown If
CANNOT_CREATE_RECORD_INSTANCE	The current record page is not scriptable or an error occurred when creating the record instance.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 var record = currentRecord.get.promise();
4 ...
5 //Add additional code

```

N/email Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/email module when you want to send email messages from within NetSuite. You can use the N/email module to send regular, bulk, and campaign email.

- [N/email Module Members](#)
- [N/email Module Script Sample](#)

N/email Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	email.send(options)	void	Client and server scripts	Sends transactional email to an individual or group of recipients and receives bounceback notifications.
	email.send.promise(options)	void	Client and server scripts	Sends transactional email asynchronously to an individual or group of recipients and receives bounceback notifications.
	email.sendBulk(options)	void	Client and server scripts	Sends bulk email (for use when a bounceback notification is not required).
	email.sendBulk.promise(options)	void	Client and server scripts	Sends bulk email asynchronously (for use when a bounceback notification is not required).
	email.sendCampaignEvent(options)	number	Client and server scripts	Sends a single "on-demand" campaign email to a specified recipient and return a campaign response ID.
	email.sendCampaignEvent.promise(options)	number	Client and server scripts	Sends a single "on-demand" campaign email asynchronously to a specified recipient and return a campaign response ID.

N/email Module Script Sample

The following script sample demonstrates how to use the features of the N/email module.



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to send an email with an attachment.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/email', 'N/record', 'N/file'], function(email, record, file) {

```

```

6   function sendEmailWithAttachment() {
7     var senderId = -5;
8     var recipientEmail = 'notify@myCompany.com';
9     var timeStamp = new Date().getUTCMilliseconds();
10
11     var recipient = record.create({
12       type: record.Type.CUSTOMER,
13       isDynamic: true
14     });
15     recipient.setValue({
16       fieldId: 'subsidiary',
17       value: '1'
18     });
19     recipient.setValue({
20       fieldId: 'companyname',
21       value: 'Test Company' + timeStamp
22     });
23     recipient.setValue({
24       fieldId: 'email',
25       value: recipientEmail
26     });
27
28     var recipientId = recipient.save();
29     var fileObj = file.load({
30       id: 88
31     });
32
33     email.send({
34       author: senderId,
35       recipients: recipientId,
36       subject: 'Test Sample Email Module',
37       body: 'email body',
38       attachments: [fileObj],
39       relatedRecords: {
40         entityId: recipientId,
41         customRecord: {
42           id: recordId,
43           recordType: recordTypeId // An integer value
44         }
45       }
46     });
47
48     sendEmailWithAttachment();
49   });
50 });

```

email.send(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends email to an individual or group of recipients and receives bounceback notifications. A maximum of 10 recipients (recipient + cc + bcc) is allowed. The total message size (including attachments) must be 20MB or less. The size of each individual attachment cannot exceed 10MB.
Returns	void
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/email Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.author	number	required	Internal ID of the email sender. To find the internal ID of the send in the UI, go to Lists > Employees.	2015.2
options.body	string	required	Contents of the outgoing message. SuiteScript formats the body of the email in either plain text or HTML. If HTML tags are present, the message is formatted as HTML. Otherwise, the message is formatted in plain text. To display XML as plain text, use an HTML <pre> tag around the XML.	2015.2
options.recipients	number string number[] number[]	required	The internal ID or email address of the recipient(s). For multiple recipients, use an array of internal IDs or email addresses. You can use an array that contains a combination of internal IDs and email addresses. A maximum of 10 recipients (recipient + cc + bcc) is allowed.	2015.2
			Note: Only the first recipient displays on the Communication tab (under the Recipient column). To view all recipients, click View to open the Message record.	
options.subject	string	required	Subject of the outgoing message.	2015.2
options.attachments	file.File	optional	Email file attachments. You can send multiple attachments of any media type. An individual attachment must not exceed 10MB and the total message size must be 20MB or less.	2015.2
			Note: Supported for server scripts only.	
options.bcc	number string number[] number[]	optional	The internal ID or email address of the recipient(s) to blind copy. For multiple recipients, use an array of internal IDs or email addresses. You can use an array that contains a combination of internal IDs and email addresses.	2015.2

Parameter	Type	Required / Optional	Description	Since
			A maximum of 10 recipients (recipient + cc + bcc) is allowed.	
options.cc	number string number[] number[][]	optional	The internal ID or email address of the secondary recipient(s) to copy. For multiple recipients, use an array of internal IDs or email addresses. You can use an array that contains a combination of internal IDs and email addresses. A maximum of 10 recipients (recipient + cc + bcc) is allowed.	2015.2
options.isInternalOnly	boolean	optional	If true, the Message record is not visible to an external Entity (for example, a customer or contact). The default value is false.	2015.2
options.relatedRecords	Object	optional	Object that contains key/value pairs to associate (attach) the Message record with related records (i.e., transaction, activity, entity, and custom records). See the RelatedRecords table for more information.	2015.2
options.replyTo	string	optional	The email address that appears in the reply-to header. You can use either a single external email address or a generic email address created by the Email Capture Plug-in.	2015.2

RelatedRecords



Note: The relatedRecords parameter is a JavaScript object.

Represents the NetSuite records to which an email Message record should be attached. You can associate the sent email with an array of internal records using key/value pairs.

There can be multiple related records, but only one of each parameter (each parameter represents applicable record types).

Parameter	Type	Required / Optional	Description	Since
activityId	number	optional	The Activity record to attach the Message record to. Use for Case and Campaign record types.	2015.2
customRecord	Object	optional	The custom record to attach the Message record to. For custom records you must specify both the record ID and the record type ID. The custom record is linked by using a nested JavaScript object.	2015.2

Parameter	Type	Required / Optional	Description	Since
customRecord.id	number	optional	The instance ID for the custom record to attach the Message record to. Note: If you use this parameter, <code>customRecord.recordType</code> is required.	2015.2
customRecord.recordType	string	optional	The integer ID for the custom record type to attach the Message record to. This ID is shown as part of the record's URL. For example: <code>/custrecordentry.n1?rectype=2&id=56</code> .	2015.2
entityId	number	optional	The Entity record to attach the Message record to. Use for all Entity record types (for example, customer, contact).	2015.2
transactionId	number	optional	The Transaction record to attach the Message record to. Use for transaction and opportunity record types.	2015.2

Errors

Error Code	Message	Thrown If
SSS_AUTHOR_MUST_BE_EMPLOYEE	The author internal id or email must match an employee.	The author internal ID or email address doesn't match an employee.
SSS_INVALID_TO_EMAIL	One or more recipient emails are not valid.	A recipient's email address is invalid.
SSS_INVALID_CC_EMAIL	One or more cc emails are not valid.	An email address specified in the <code>options.cc</code> parameter is invalid.
SSS_INVALID_BCC_EMAIL	One or more bcc emails are not valid.	An email address specified in the <code>options.bcc</code> parameter is invalid
SSS_MAXIMUM_NUMBER_RECIPIENTS_EXCEEDED	You may have a maximum number of 10 recipients when <code>"notifySenderOnBounce"</code> is set to true.	The total number of recipients (recipient + cc + bcc) exceeds 10.
SSS_MISSING_REQD_ARGUMENT	{method name}: Missing a required argument: {param name}	A required parameter is missing.
WRONG_PARAMETER_TYPE	Wrong parameter type: {param name} is expected as {param type}.	A parameter's type is incorrect.

Error Code	Message	Thrown If
SSS_FILE_CONTENT_SIZE_EXCEEDED	The file content you are attempting to access exceeds the maximum allowed size of 10MB.	An attachment exceeds the 10 MB file size limit.
ATTACH_SIZE_EXCEEDED	This message exceeds the limit of 20 MB. Please reduce the size of the message and its attachments and try again. Note: Files can be larger when attached due to encoding.	The size of the attachments exceeds the limit.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/email Module Script Sample](#).

```

1 //Add additional code
2 .
3 var senderId = -5;
4 var recipientEmail = 'notify@myCompany.com';
5 var timeStamp = new Date().getUTCMilliseconds();
6 var recipientId = 12;
7 var fileObj = file.load({
8   id: 88
9 });
10 email.send({
11   author: senderId,
12   recipients: recipientId,
13   subject: 'Test Sample Email Module',
14   body: 'email body',
15   attachments: [fileObj],
16   relatedRecords: {
17     entityId: recipientId,
18     customRecord:{
19       id:recordId,
20       recordType: recordTypeId //an integer value
21     }
22   }
23 });
24 ...
25 //Add additional code

```

email.send.promise(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends email asynchronously to an individual or group of recipients and receives bounceback notifications. i Note: The parameters and errors thrown for this method are the same as those for email.send(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	email.send(options)
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units

Module	N/email Module
Since	2015.2

email.sendBulk(options)

<p>Note: The content in this help topic pertains to SuiteScript 2.0.</p>	
Method Description	<p>Sends bulk email (for use when a bounceback notification is not required). A maximum of 10 recipients (recipient + cc + bcc) is allowed. The total message size (including attachments) must be 20MB or less. The size of each individual attachment cannot exceed 10MB.</p> <p>Note: This API normally uses a bulk email server to send messages. If you need to increase the successful delivery rate of an email, use email.send(options) so that a transactional email server is used.</p>
Returns	void
Supported Script Types	<p>Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	10 units
Module	N/email Module
Since	2015.2

Parameters

<p>Note: The options parameter is a JavaScript object.</p>				
Parameter	Type	Required / Optional	Description	Since
options.author	number	required	<p>Internal ID of the email sender. To find the internal ID of the send in the UI, go to Lists > Employees.</p>	2015.2
options.body	string	required	<p>Contents of the outgoing message. SuiteScript formats the body of the email in either plain text or HTML. If HTML tags are present, the message is formatted as HTML. Otherwise, the message is formatted in plain text. To display XML as plain text, use an HTML <pre> tag around the XML.</p>	2015.2
options.recipients	number string number[] string[]	required	<p>The internal ID or email address of the recipient. For multiple recipients, use an array of internal IDs or email addresses. You can use an array that contains a combination of internal IDs and email addresses.</p>	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p>A maximum of 10 recipients (recipient + cc + bcc) is allowed.</p> <p>Note: Only the first recipient displays on the Communication tab (under the Recipient column). To view all recipients, click View to open the Message record.</p>	
options.subject	string	required	Subject of the outgoing message.	2015.2
options.attachments	file.File	optional	<p>The email file attachments.</p> <p>You can send multiple attachments of any media type.</p> <p>An individual attachment must not exceed 10MB and the total message size must be 20MB or less.</p> <p>Note: Supported for server scripts only.</p>	2015.2
options.bcc	number number[] string string[]	optional	<p>The internal ID or email address of the recipient to blind copy.</p> <p>For multiple recipients, use an array of internal IDs or email addresses. You can use an array that contains a combination of internal IDs and email addresses.</p> <p>A maximum of 10 recipients (recipient + cc + bcc) is allowed.</p>	2015.2
options.cc	number number[] string string[]	optional	<p>The internal ID or email address of the secondary recipient to copy.</p> <p>For multiple recipients, use an array of internal IDs or email addresses. You can use an array that contains a combination of internal IDs and email addresses.</p> <p>A maximum of 10 recipients (recipient + cc + bcc) is allowed.</p>	2015.2
options.isInternalOnly	boolean	optional	<p>If true, the Message record is not visible to an external Entity (for example, a customer or contact).</p> <p>The default value is false.</p>	2015.2
options.relatedRecords	Object	optional	<p>Object that contains key/value pairs to associate (attach) the Message record with related records (i.e., transaction, activity, entity, and custom records).</p> <p>See the RelatedRecords table for more information.</p>	2015.2
options.replyTo	string	optional	The email address that appears in the reply-to header.	2015.2

Parameter	Type	Required / Optional	Description	Since
			You can use either a single external email address or a generic email address created by the Email Capture Plug-in.	

RelatedRecords



Note: The relatedRecords parameter is a JavaScript object.

Represents the NetSuite records to which an email Message record should be attached. You can associate the sent email with an array of internal records using key/value pairs.

There can be multiple related records, but only one of each parameter (each parameter represents applicable record types).

Parameter	Type	Required / Optional	Description	Since
activityId	number	optional	The Activity record to attach the Message record to. Use for Case and Campaign record types.	2015.2
entityId	number	optional	The Entity record to attach the Message record to. Use for all Entity record types (for example, customer, contact).	2015.2
customRecord	Object	optional	The custom record to attach the Message record to. For custom records you must specify both the record ID and the record type ID. The custom record is linked by using a nested JavaScript object.	2015.2
customRecord.id	number	optional	The instance ID for the custom record to attach the Message record to.	2015.2
<p> Note: If you use this parameter, customRecord.recordType is required.</p>				
customRecord.recordType	string	optional	The integer ID for the custom record type to attach the Message record to. This ID is shown as part of the record's URL. For example: /custrecordentry.n1?rectype=2&id=56.	2015.2
<p> Note: If you use this parameter, customRecord.id is required.</p>				
transactionId	number	optional	The Transaction record to attach the Message record to. Use for transaction and opportunity record types.	2015.2

Errors

Error Code	Message	Thrown If
SSS_AUTHOR_MUST_BE_EMPLOYEE	The author internal id or email must match an employee.	The author internal ID or email address doesn't match an employee.
SSS_INVALID_TO_EMAIL	One or more recipient emails are not valid.	A recipient's email address is invalid.
SSS_INVALID_CC_EMAIL	One or more cc emails are not valid.	An email address specified in the options.cc parameter is invalid.
SSS_INVALID_BCC_EMAIL	One or more bcc emails are not valid.	An email address specified in the options.bcc parameter is invalid
SSS_MAXIMUM_NUMBER_RECIPIENTS_EXCEEDED	You may have a maximum number of 10 recipients when \"notifySenderOnBounce\" is set to true.	The total number of recipients (recipient + cc + bcc) exceeds 10.
SSS_MISSING_REQD_ARGUMENT	{method name}: Missing a required argument: {param name}	A required parameter is missing.
WRONG_PARAMETER_TYPE	Wrong parameter type: {param name} is expected as {param type}.	A parameter's type is incorrect.
SSS_FILE_CONTENT_SIZE_EXCEEDED	The file content you are attempting to access exceeds the maximum allowed size of 10.0 MB.	An attachment exceeds the 10 MB file size limit.
ATTACH_SIZE_EXCEEDED	This message exceeds the limit of 20 MB. Please reduce the size of the message and its attachments and try again. Note: Files can be larger when attached due to encoding.	The size of the attachments exceeds the limit.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/email Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var recipientEmails = [
4   'msample@netsuite.com',
5   'jdoe@netsuite.com',
6   'awolfe@netsuite.com',
7   'htest@netsuite.com'
8 ];
9 email.sendBulk({
10   author: -5,
11   recipients: recipientEmails,
12   subject: 'Order Status',
13   body: 'Your order has been completed.',
14   replyTo: 'accounts@netsuite.com'
15 });
16 ...
17 //Add additional code

```

email.sendBulk.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends bulk email asynchronously (for use when a bounceback notification is not required).
	<p> Note: The parameters and errors thrown for this method are the same as those for email.sendBulk(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	email.sendBulk(options)
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/email Module
Since	2015.2

email.sendCampaignEvent(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends a single “on-demand” campaign email to a specified recipient and return a campaign response ID to track the email. This is used for lead nurturing campaigns (drip marketing email). Email (campaignemail) sublists are not supported. The campaign must use a Lead Nurturing (campaigndrip) sublist.
	<p> Note: This API normally uses a bulk email server to send messages. If you need to increase the successful delivery rate of an email, use email.send(options) so that a transactional email server is used.</p>
Returns	A campaign response ID (tracking code) as number If the email fails to send, the value returned is -1.
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/email Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.campaignEventId	number	required	<p>The internal ID of the campaign event.</p> <p> Note: This campaign must use a Lead Nurturing (campaignndrip) sublist. For more information on Lead Nurturing campaigns, see the help topic Lead Nurturing Campaigns.</p>	2015.2
options.recipientId	number	required	<p>The internal ID of the recipient.</p> <p> Note: The recipient's record must contain an email address.</p>	2015.2

Errors

Error Code	Message	Thrown If
SSS_AUTHOR_MUST_BE_EMPLOYEE	The author internal id or email must match an employee.	The author internal ID or email address doesn't match an employee.
SSS_INVALID_TO_EMAIL	One or more recipient emails are not valid.	A recipient's email address is invalid.
SSS_INVALID_CC_EMAIL	One or more cc emails are not valid.	An email address specified in the options.cc parameter is invalid.
SSS_INVALID_BCC_EMAIL	One or more bcc emails are not valid.	An email address specified in the options.bcc parameter is invalid
SSS_MAXIMUM_NUMBER_RECIPIENTS_EXCEEDED	You may have a maximum number of 10 recipients when \"notifySenderOnBounce\" is set to true.	The total number of recipients (recipient + cc + bcc) exceeds 10.
SSS_MISSING_REQD_ARGUMENT	{method name}: Missing a required argument: {param name}	A required parameter is missing.
WRONG_PARAMETER_TYPE	Wrong parameter type: {param name} is expected as {param type}.	A parameter's type is incorrect.
SSS_FILE_CONTENT_SIZE_EXCEEDED	The file content you are attempting to access exceeds the maximum allowed size of 10.0 MB.	An attachment exceeds the 10 MB file size limit.
ATTACH_SIZE_EXCEEDED	This message exceeds the limit of 20 MB. Please reduce the size of the message and its attachments	The size of the attachments exceeds the limit.

Error Code	Message	Thrown If
	and try again. Note: Files can be larger when attached due to encoding.	

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/email Module Script Sample](#).



Note: The following script includes sample values for some fields. You must replace these values with values from your NetSuite account in order for the code to execute successfully.

```

1 // Add additional code
2 ...
3 // Create a campaign record in dynamic mode so all field values can be dynamically sourced.
4 var campaign1 = record.create({
5     type: record.Type.CAMPAIGN,
6     isDynamic: true
7 });
8 campaign1.setValue({
9     fieldId: 'title',
10    value: 'Sample Lead Nurturing Campaign'
11 });
12 // set values on the Lead Nurturing (campaigndrip) sublist
13 campaign1.selectNewLine({
14     sublistId: 'campaigndrip'
15 });
16 // 4 is a sample ID representing an existing marketing campaign
17 campaign1.setCurrentSublistValue({
18     sublistId: 'campaigndrip',
19     fieldId: 'template',
20     value: 4
21 });
22 campaign1.setCurrentSublistValue({
23     sublistId: 'campaigndrip',
24     fieldId: 'title',
25     value: 'Sample Lead Nurturing Event'
26 });
27 // 1 is a sample ID representing an existing subscription
28 campaign1.setCurrentSublistValue({
29     sublistId: 'campaigndrip',
30     fieldId: 'subscription',
31     value: 1
32 });
33
34 // 2 is a sample ID representing an existing channel
35 campaign1.setCurrentSublistValue({
36     sublistId: 'campaigndrip',
37     fieldId: 'channel',
38     value: 2
39 });
40
41 // 1 is a sample ID representing an existing promocode
42 campaign1.setCurrentSublistValue({
43     sublistId: 'campaigndrip',
44     fieldId: 'promocode',
45     value: 1
46 });
47 campaign1.commitLine({
48     sublistId: 'campaigndrip'
49 });
50
51 // Submit the record var
52 campaign1Key = campaign1.save();
53

```

```

54 // Load the campaign record you just created. Determine the internal ID of the campaign event to the variable campaign2_campaigndrip_internalid_1.
55 var campaign2 = record.load({
56   type: record.Type.CAMPAIGN,
57   id : campaign1Key,
58   isDynamic: true
59 });
60 var campaign2_campaigndrip_internalid_1 = campaign2.getSublistValue({
61   sublistId: 'campaigndrip',
62   fieldId: 'internalid',
63   line: 1
64 });
65 // 142 is a sample ID representing the ID of a recipient with a valid email address
66 var campaignResponseId = email.sendCampaignEvent({
67   campaignEventId: campaign2_campaigndrip_internalid_1,
68   recipientId: 142
69 });
70 ...
71 //Add additional code

```

email.sendCampaignEvent.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends a single “on-demand” campaign email asynchronously to a specified recipient and return a campaign response ID to track the email. This is used for lead nurturing campaigns (drip marketing email). If the email fails to send, the value returned is -1.
	 Note: The parameters and errors thrown for this method are the same as those for email.sendCampaignEvent(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	email.sendCampaignEvent(options)
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/email Module
Since	2015.2

N/encode Module

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

This module exposes string encoding and decoding functionality. Load the N/encode module when you want to convert a string to another type of encoding.

- [N/encode Module Members](#)
- [N/encode Module Script Samples](#)

N/encode Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	encode.convert(options)	string	Server-side scripts	Converts a string to another type of encoding and returns the re-encoded string.
Enum	encode.Encoding	enum	Server-side scripts	Holds the string values for supported encoding specifications.

N/encode Module Script Samples

The following script samples demonstrate how to use the features of the N/encode module.

Sample 1: Convert a string to a different encoding

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following script shows how to convert a string to a different encoding.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/encode'], function(encode) {
6     function convertStringToDifferentEncoding() {
7         var stringInput = "TÃƒÂ©st StriÃƒÂ¢g Input";
8         var base64EncodedString = encode.convert({
9             string: stringInput,
10            inputEncoding: encode.Encoding.UTF_8,
11            outputEncoding: encode.Encoding.BASE_64
12        });
13        var hexEncodedString = encode.convert({
14            string: stringInput,
15            inputEncoding: encode.Encoding.UTF_8,
16            outputEncoding: encode.Encoding.HEX
17        });
18    }
19
20    convertStringToDifferentEncoding();
21 });

```

encode.convert(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description

Converts a string to another type of encoding.

Returns	The re-encoded string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/encode Module
Since	2015.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.string	string	required	The string to encode.	2015.1
options.inputEncoding	string	required	The encoding used on the input string. The default value is UTF_8. Use the encode.Encoding to set the value.	2015.1
options.outputEncoding	string	required	The encoding to apply to the output string. The default value is UTF_8. Use the encode.Encoding to set the value.	2015.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [encode Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var hexEncodedString = encode.convert({
4     string: stringInput,
5     inputEncoding: encode.Encoding.UTF_8,
6     outputEncoding: encode.Encoding.HEX
7 });
8 ...
9 //Add additional code

```

encode.Encoding

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for the supported character set encoding. This enum is used to set the value of inputEncoding and outputEncoding parameters that are members of the N/crypto Module or N/encode Module .
-------------------------	---

	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/encode Module
Since	2015.1

Values

- UTF_8
- BASE_16
- BASE_32
- BASE_64
- BASE_64_URL_SAFE
- HEX

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [encode Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var reencoded = encode.convert({
4     string: LOREM_IPS,
5     inputEncoding: encode.Encoding.BASE_64,
6     outputEncoding: encode.Encoding.UTF_8
7 });
8 ...
9 //Add additional code

```

N/error Module

Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/error module when you want to create your own custom SuiteScript errors. Use these custom errors in try-catch statements to abort script execution. Your script can also include logic to throw custom SuiteScript errors after they are created. This module does not provide functionality to throw custom errors.

- [N/error Module Members](#)
- [SuiteScriptError Object Members](#)
- [UserEventError Object Members](#)
- [N/error Module Script Samples](#)

Also refer to the [N/log Module](#) for additional error logging capabilities.

N/error Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	error.SuiteScriptError	Object	Server scripts that are not user event scripts	Encapsulates a custom SuiteScript error for any server script type that is not a user event script.
	error.UserEventError	Object	User event scripts	Encapsulates a custom SuiteScript error for a user event script.
Method	error.create(options)	error.SuiteScriptError	Server scripts	Creates a new error.SuiteScriptError object.
Enum	error.Type	enum	Server scripts	Holds the string values for error types.

SuiteScriptError Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	SuiteScriptError.cause	string (read-only)	Server scripts that are not user event scripts	Cause of the error.
	SuiteScriptError.id	string (read-only)	Server scripts that are not user event scripts	Error ID that is automatically generated when a new error is created.
	SuiteScriptError.message	string (read-only)	Server scripts that are not user event scripts	Error message text displayed in the Details column of the Execution Log.
	SuiteScriptError.name	string (read-only)	Server scripts that are not user event scripts	User-defined error code.
	SuiteScriptError.notifyOff	boolean (read-only)	Server scripts that are not user event scripts	Suppresses email notification when set to true.
	SuiteScriptError.stack	Array of strings (read-only)	Server scripts that are not user event scripts	List of method calls that the script is executing when the error is thrown.
	SuiteScriptError.type	error.Type (read-only)	Server scripts that are not user event scripts	Error types.

UserEventError Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	UserEventError.cause	string (read-only)	User event scripts	Cause of the error.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	UserEventError.eventType	string (read-only)	User event scripts	User event type (beforeLoad, beforeSubmit, afterSubmit)
	UserEventError.id	string (read-only)	User event scripts	Error ID that is automatically generated when a new error is created.
	UserEventError.message	string (read-only)	User event scripts	Error message text displayed in the Details column of the Execution Log.
	UserEventError.name	string (read-only)	User event scripts	User-defined error code.
	UserEventError.notifyOff	boolean (read-only)	User event scripts	Suppresses email notification when set to true.
	UserEventError.recordId	string (read-only)	User event scripts	Internal ID of the submitted record that triggered the script. This property only holds a value when the error is thrown by an afterSubmit user event.
	UserEventError.stack	Array of strings (read-only)	User event scripts	List of method calls that the script is executing when the error is thrown.
	UserEventError.type	error.Type (read-only)	User event scripts	Error types.

N/error Module Script Samples

The following script samples show two ways of creating a custom error using the N/error module.

Sample 1: Create an error

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to create a custom error. If this sample code is used in a server script that is not a user event script, errorObj will be an error.SuiteSError object. If this sample code is used in a user event script, errorObj will be an error.UserEventError object.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/error'], function(error) {
6     function createError() {
7         var myCustomError = error.create({
8             name: 'MY_ERROR_CODE',
9             message: 'My custom error details',
10            notifyOff: true
11        });
12    }
13
14    createError();

```

15 | }));

Sample 2: Create an error based on a condition

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to conditionally create and throw a custom error. If this sample code is used in a server script that is not a user event script, errorObj will be an error.SuiteScriptError object. If this sample code is used in a user event script, errorObj will be an error.UserEventError object.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/error'], function(error) {
6      function showError() {
7          var someVariable = false;
8
9          if (!someVariable) {
10              var myCustomError = error.create({
11                  name: 'WRONG_PARAMETER_TYPE',
12                  message: 'Wrong parameter type selected.',
13                  notifyOff: false
14              });
15
16              // This will write 'Error: WRONG_PARAMETER_TYPE Wrong parameter type selected' to the log
17              log.error('Error: ' + myCustomError.name , myCustomError.message);
18              throw errObj;
19          }
20      }
21
22      showError();
23  });

```

error.SuiteScriptError

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a custom SuiteScript error for any server script type that is not a user event script. Use this object in a try-catch statement to abort script execution. Create a new custom error with the error.create(options) method. The error.create(options) method returns error.SuiteScriptError when it is called in any server script that is not a user event script. For a complete list of this object's methods and properties, see SuiteScriptError Object Members .
Supported Script Types	All server scripts that are not user event scripts. For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 // include this code in a server script that is not a user event script to create an error.SuiteScriptError object
4 var SScustom_error = error.create({
5   name: 'MY_ERROR_CODE',
6   message: 'my custom SuiteScriptError details',
7   notifyOff: false
8 });
9 ...
10 //Add additional code

```

SuiteScriptError.cause



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The cause of the error.
Type	string (read-only)
Supported Script Types	All server scripts that are not user event scripts. For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 var SScustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom SuiteScriptError details',
6   cause: 'the cause of the error',
7   notifyOff: false
8 });
9 log.debug("Error cause: " + SScustom_error.cause); //Error cause:' followed by the cause will be logged
10 ...
11 //Add additional code

```

SuiteScriptError.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Error ID that is automatically generated when a new error is created.
-----------------------------	---

Type	string (read-only)
Supported Script Types	All server scripts that are not user event scripts. For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var SScustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom SuiteScriptError details',
6   notifyOff: false
7 });
8 log.debug("Error ID: " + SScustom_error.id); //id is system generated
9                                         //Error ID: followed by the id will be logged
10 ...
11 //Add additional code

```

SuiteScriptError.message

ℹ Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Error message text displayed in the Details column of the Execution Log.
Type	string (read-only)
Supported Script Types	All server scripts that are not user event scripts. For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var SScustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom SuiteScriptError details',
6   notifyOff: false
7 });
8 log.debug("Error Message: " + SScustom_error.message); //Error Message: my custom SuiteScriptError details' will be logged
9 ...
10 //Add additional code

```

SuiteScriptError.name



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	User-defined error code.
Type	string (read-only)
Supported Script Types	All server scripts that are not user event scripts. For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var SScustom_error = error.create({
4     name: 'MY_ERROR_CODE',
5     message: 'my custom SuiteScriptError details',
6     notifyOff: false
7 });
8 log.debug("Error Code: " + SScustom_error.name); // 'Error Code: MY_ERROR_CODE' will be logged
9 ...
10 //Add additional code

```

SuiteScriptError.notifyOff



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Suppresses email notification when set to true.
Type	string (read-only)
Supported Script Types	All server scripts that are not user event scripts. For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```
1 //Add additional code
```

```

2 ...
3 var SScustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom SuiteScriptError details',
6   notifyOff: false
7 });
8 log.debug("Error NotifyOff: " + SScustom_error.notifyOff); // 'Error NotifyOff:false' will be logged
9 ...
10 //Add additional code

```

SuiteScriptError.stack



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	List of method calls that the script is executing when the error is thrown. The most recently executed method is listed at the top of the list.
Type	Array of strings (read-only)
Supported Script Types	All server scripts that are not user event scripts. For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 var SScustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom SuiteScriptError details',
6   notifyOff: false
7 });
8 log.debug("Error Stack: " + SScustom_error.stack); // stack is set by the system
9                                     // 'Error Stack:' followed by the stack will be logged
10 ...
11 //Add additional code

```

SuiteScriptError.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Error types. Use values defined in error.Type .
Type	string (read-only)
Supported Script Types	All server scripts that are not user event scripts. For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/error Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 var SScustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom SuiteScriptError details',
6   notifyOff: false
7 });
8 log.debug("Error Type: " + SScustom_error.type); // 'Error Type:' followed by the type will be logged
9 ...
10 //Add additional code

```

error.UserEventError

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a custom SuiteScript error for a user event script. Use this object in a try-catch statement to abort script execution. Create a new custom error with the error.create(options) method. The error.create(options) method returns <code>error.UserEventError</code> when it is called in a user event script. Note: When <code>error.create(options)</code> is called in a server script that is not a user event script, it returns <code>error.SuiteScriptError</code> .
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Module	N/error Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 // Include this code in a user event script to create an error.UserEventError object
4 var UEventError = error.create({

```

```

5   name: 'MY_ERROR_CODE',
6   message: 'my custom UserEventError details',
7   notifyOff: false
8 });
9 ...
10 //Add additional code

```

UserEventError.cause



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The cause of the error.
Type	string (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var UEcustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom UserEventError details',
6   cause: 'the cause of the error',
7   notifyOff: false
8 });
9 log.debug("Error cause: " + UEcustom_error.cause); // 'Error cause:' followed by the cause will be logged
10 ...
11 //Add additional code

```

UserEventError.eventType



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	User event type (beforeLoad, beforeSubmit, or afterSubmit)
Type	string (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 var UEcustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom UserEventError details',
6   notifyOff: false
7 });
8 log.debug("User Event Type: " + UEcustom_error.eventType); // eventType is set by the system
9                                         // 'User Event Type:' followed by the eventType will be logged
10 ...
11 //Add additional code

```

UserEventError.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Error ID that is automatically generated when a new error is created.
Type	string (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 var UEcustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom UserEventError details',
6   notifyOff: false
7 });
8 log.debug("Error ID: " + UEcustom_error.id); // id is system generated
9                                         // Error ID: 'followed by the id will be logged'
10 ...
11 //Add additional code

```

UserEventError.message



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Error message text displayed in the Details column of the Execution Log.
-----------------------------	--

Type	string (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var UEventError = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom UserEventError details',
6   notifyOff: false
7 });
8 log.debug("Error Message: " + UEventError.message); //Error Message: my custom UserEventError details' will be logged
9 ...
10 //Add additional code

```

UserEventError.name



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	User-defined error code.
Type	string (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var UEventError = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom UserEventError details',
6   notifyOff: false
7 });
8 log.debug("Error Code: " + UEventError.name); //Error Code: MY_ERROR_CODE' will be logged
9 ...
10 //Add additional code

```

UserEventError.notifyOff

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Suppresses email notification when set to true.
Type	string (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var UECustom_Error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom SuiteScriptError details',
6   notifyOff: false
7 });
8 log.debug("Error NotifyOff: " + UECustom_Error.notifyOff); // 'Error NotifyOff: false' will be logged
9 ...
10 //Add additional code

```

UserEventError.recordId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID of the submitted record that triggered the script. This property only holds a value when the error is thrown by an afterSubmit user event script.
Type	string (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Module	N/error Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var UECustom_Error = error.create({

```

```

4   name: 'MY_ERROR_CODE',
5   message: 'my custom UserEventError details',
6   notifyOff: false
7 });
8 log.debug("Submitted Record ID: " + UEcustom_error.recordId); //recordId is set by the system
9                                         // 'Submitted Record ID: ' followed by the recordId will be logged
10 ...
11 //Add additional code

```

UserEventError.stack



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	List of method calls that the script is executing when the error is thrown. The most recently executed method is listed at the top of the list.
Type	Array of strings (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/error Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var UEcustom_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom UserEventError details',
6   notifyOff: false
7 });
8 log.debug("Error Stack: " + UEcustom_error.stack); //stack is set by the system
9                                         // 'Error Stack: ' followed by the stack will be logged
10 ...
11 //Add additional code

```

UserEventError.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Error types. Use values defined in error.Type .
Type	string (read-only)
Supported Script Types	User event scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 var UEcumon_error = error.create({
4   name: 'MY_ERROR_CODE',
5   message: 'my custom SuiteScriptError details',
6   notifyOff: false
7 });
8 log.debug("Error Type: " + UEcumon_error.type); //Error Type: followed by the type will be logged
9 ...
10 //Add additional code

```

error.create(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new <code>error.SuiteScriptError</code> object. Use the <code>error.SuiteScriptError</code> or <code>error.UserEventError</code> object in a try-catch statement to abort script execution. Note this method creates a new error, but does not throw the error. Your script code will need to include logic to throw the error when appropriate.
Returns	An <code>error.SuiteScriptError</code> object.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/error Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object. The table below describes the name:value pairs that make up the object.

Parameter	Type	Required / Optional	Description	Since
options.message	string	required	Error message text displayed in the Details column of the Execution Log. Sets the value for the <code>SuiteScriptError.message</code> property. The default value is null.	2015.2
options.name	string	required	User-defined error code. Sets the value for the <code>SuiteScriptError.name</code> property.	2015.2
options.notifyOff	boolean	optional	Sets whether email notification is suppressed. If set to false, the system emails the users identified	2015.2

Parameter	Type	Required / Optional	Description	Since
			on the applicable script record's Unhandled Errors subtab when the error is thrown. For additional information on the Unhandled Errors subtab, see the help topic Creating a Script Record .	

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	The options.message or options.name parameter is not specified.
WRONG_PARAMETER_TYPE	One of the parameters is not specified as the correct type: <ul style="list-style-type: none"> ■ options.message must be a string value ■ options.name must be a string value ■ options.notifyOff must be a boolean value

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/error Module Script Sample.

```

1 //Add additional code
2 ...
3 // errorObj is a error.SuiteScriptError if this code is included in a server script that is not a user event script.
4 // errorObj is a error.UserEventError if this code is included in a user event script.
5
6 var custom_error = error.create({
7   name: 'MY_ERROR_CODE',
8   message: 'my custom error details',
9   notifyOff: false
10 });
11 log.debug("Error Code: " + custom_error.name); // 'Error Code: MY_ERROR_CODE' will be logged
12 ...
13 //Add additional code

```

error.Type

ⓘ Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for error types. You can use these error types and write your own corresponding error messages. This may be useful if you are developing a SuiteScript library.
	ⓘ Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/error Module

Values

Value
A_SCRIPT_IS_ATTEMPTING_TO_EDIT_THE_1_SUBLIST_THIS_SUBLIST_IS_CURRENTLY_IN_READONLY_MODE_AND_CANNOT_BE_EDITED_CALL_YOUR_NETSUITE_ADMINISTRATOR_TO_DISABLE_THIS_SCRIPT_IF_YOU_NEED_TO_SUBMIT_THIS_RECORD
AT_LEAST_ONE_EXPRESSION_IS_NEEDED
BUTTONS_MUST_INCLUDE_BOTH_A_LABEL_AND_VALUE
CAN_REFERENCE_ONLY_PERSISTED_DATASET
CANNOT_CREATE_RECORD_DRAFT_OF_EXISTING_RECORD
CANNOT_CREATE_RECORD_INSTANCE
CANNOT_DETERMINE_TYPE_FOR_ALIAS
CANNOT_DETERMINE_VALUE_FOR_ALIAS
CANNOT_RESUBMIT_SUBMITTED_ASYNC_QUERY_TASK
CANNOT_RESUBMIT_SUBMITTED_ASYNC_SEARCH_TASK
CANNOT_RESUBMIT_SUBMITTED_ASYNC_SUITEQL_TASK
COLOR_VALUE_MUST_BE_6_HEXADECIMAL_DIGITS_OF_THE_FORM_RRGGBB__EXAMPLE_FF0000_FOR_RED
CREDIT_CARD_NUMBER_IS_NOT_VALID__PLEASE_CHECK_THAT_ALL_DIGITS_WERE_ENTERED_CORRECTLY
CREDIT_CARD_NUMBER_MUST_CONTAIN_ONLY_DIGITS
CREDIT_CARD_NUMBERS_MUST_CONTAIN_BETWEEN_13_AND_20_DIGITS
DATASET_NAME_IS_MISSING
EACH_VIEW_MUST_HAVE_A_NAME
EACH_VIEW_MUST_HAVE_AN_ID
EMPTY_KEY_NOT_ALLOWED
FAILED_AN_UNEXPECTED_ERROR_OCCURRED
FIELD_1_ALREADY_CONTAINS_A_SUBRECORD_YOU_CANNOT_CALL_CREATESUBRECORD
FIELD_1_CANNOT_BE_EMPTY
FIELD_1_IS_NOT_A_SUBRECORD_FIELD
FIELD_MUST_CONTAIN_A_VALUE
FORM_VALIDATION_FAILED_YOU_CANNOT_CREATE_THIS_SUBRECORD
FORM_VALIDATION_FAILED_YOU_CANNOT_SUBMIT_THIS_RECORD
HISTORY_IS_ONLY_AVAILABLE_FOR_THE_LAST_30_DAYS
ID_CANNOT_HAVE_MORE_THAN_N_CHARACTERS
IDENTIFIERS_CAN_CONTAIN_ONLY_DIGITS_ALPHABETIC_CHARACTERS_OR_WITH_NO_SPACES
INVALID_AGGREGATE_TYPE
INVALID_AGGREGATION

Value
INVALID_ALGORITHM
INVALID_ASPECT_TYPE
INVALID_CERTIFICATE_TYPE
INVALID_CHART_TYPE
INVALID_COLUMN_FOR_SORTING
INVALID_CONFIGURATION_UNABLE_TO_CHANGE_REQUIRE_CONFIGURATION_FOR_1
INVALID_CONFIGURATION_UNABLE_TO_CHANGE_REQUIRE_CONFIGURATION_WITHOUT_A_CONTEXT
INVALID_CUSTOM_VIEW_VALUE
INVALID_DATASET_ID
INVALID_DATE_ID
INVALID_DATE_VALUE_MUST_BE_1
INVALID_DATE_VALUE_MUST_BE_ON_OR_AFTER_1CUTOFF_DATE
INVALID_DIRECTION_FOR_SORTING
INVALID_EMAILS_FOUND
INVALID_EXPRESSION
INVALID_FIELD_CONTEXT
INVALID_FIELD_ID
INVALID_FIELD_INDEX
INVALID_FIELD_VALUE
INVALID_FILTER_FIELD_FOR_CURRENT_VIEW
INVALID_FLD_VALUE
INVALID_FORMULA_TYPE
INVALID_HTTP_METHOD
INVALID_ID_PREFIX
INVALID_KEY_TYPE
INVALID_LOCALE
INVALID_NUMBER_MUST_BE_BETWEEN_1_AND_2
INVALID_NUMBER_MUST_BE_GREATE_THAN_1
INVALID_NUMBER_MUST_BE_LOWER_THAN_1
INVALID_NUMBER_OR_PERCENTAGE
INVALID_OPERATION
INVALID_OPERATOR
INVALID_OR_UNSUPPORTED_RECORD_TYPE_1

Value
INVALID_PAGE_INDEX
INVALID_PAGE_RANGE
INVALID_PERIOD_ADJUSTMENT
INVALID_PERIOD_CODE
INVALID_PERIOD_TYPE
INVALID_RCRD_TYPE
INVALID_RETURN_TYPE_EXPECTED_1
INVALID_SEARCH_OPERATOR
INVALID_SEARCH_TYPE
INVALID_SIGNATURE
INVALID_SIGNATURE_TAG
INVALID_SORT
INVALID_SORT_LOCALE
INVALID_STACKING_TYPE
INVALID_SUBRECORD_MERGE
INVALID_SUBRECORD_REFERENCE
INVALID_SUITEAPP_APPLICATION_ID
INVALID_TASK_TYPE
INVALID_TOTAL_LINE
INVALID_URL_SPACES_ARE_NOT_ALLOWED_IN_THE_URL
INVALID_URL_URL_MUST_START_WITH_HTTP_HTTPS_FTP_OR_FILE
INVALID_WORKBOOK_ID
MUTUALLY_EXCLUSIVE_ARGUMENTS
NAME_CANNOT_BE_EMPTY
NAME_CANNOT_HAVE_MORE_THAN_N_CHARACTERS
NEITHER_ARGUMENT_DEFINED
NO_ASPECTS_DEFINED
NO_CHILDREN_DEFINED
NO_COLUMN_DEFINED
NO_DIMENSION_ITEM_DEFINED
NO_ELEMENTS_DEFINED
NO_SORT_BY_DEFINED
NON_KATAKANA_DATA_FOUND

Value
NOT_SUPPORTED_ON_CURRENT_SUBRECORD
NOTICE_THE_CREDIT_CARD_APPEARS_TO_BE_INCORRECT
OPERATION_IS_NOT_ALLOWED
OPERATOR_ARITY_MISMATCH
PASSWORD_CANNOT_HAVE_MORE_THAN_N_CHARACTERS
PHONE_NUMBER_SHOULD_HAVE_SEVEN_DIGITS_OR_MORE
PLEASE_ENTER_A_VALID_FROM_START_DATE_IN_MMYYYY_FORMAT
PLEASE_ENTER_AN_EXPIRATION_DATE_IN_MMYYYY_FORMAT
PLEASE_INCLUDE_THE_AREA_CODE_FOR_PHONE_NUMBER
PROPERTY_VALUE_CONFLICT
READ_ONLY_PROPERTY
RELATIONSHIP_ALREADY_USED
SCRIPT_EXECUTION_USAGE_LIMIT_EXCEEDED
SELECT_OPTION_ALREADY_PRESENT
SELECT_OPTION_NOT_FOUND
SIGNATURE_VERIFICATION_FAILED
SSS_ARGUMENT_DISCREPANCY
SSS_DUPLICATE_ALIAS
SSS_INVALID_ACTION_ID
SSS_INVALID_API_USAGE
SSS_INVALID_COUNTRY_ID
SSS_INVALID_CURRENCY_ID
SSS_INVALID_FORMAT_TYPE
SSS_INVALID_GETSELECTOPTION_FILTER_OPERATOR
SSS_INVALID_MACRO_ID
SSS_INVALID_READ_SIZE
SSS_INVALID_SEARCH_RESULT_INDEX
SSS_INVALID_SEGMENT_SEPARATOR
SSS_INVALID_SRCH_OPERATOR
SSS_INVALID sublist
SSS_INVALID sublist_OPERATION
SSS_INVALID_TYPE_ARG
SSS_INVALID_UI_OBJECT_TYPE

Value
SSS_INVALID_URL
SSS_METHOD_IS_ONLY_ALLOWED_FOR_MATRIX_FIELD
SSS_METHOD_IS_ONLY_ALLOWED_FOR_MULTISELECT_FIELD
SSS_METHOD_IS_ONLY_FOR_SELECT_FIELD
SSS_MISSING_ALIAS
SSS_MISSING_REQD_ARGUMENT
SSS_NOT_YET_SUPPORTED
SSS_RECORD_DOES_NOT_SATISFY_CONDITION
SSS_RECORD_TYPE_MISMATCH
SSS_SEARCH_FOR_EACH_LIMIT_EXCEEDED
SSS_SEARCH_RESULT_LIMIT_EXCEEDED
SSS_SUBLIST_DOESNT_SUPPORT_MOVING_LINES
SSS_TAG_CANNOT_BE_EMPTY
SSS_TAX_REGISTRATION_REQUIRED
SSS_UNSUPPORTED_METHOD
TABLE_DOES_NOT_EXIST
THAT_RECORD_IS_NOT_EDITABLE
THE_FIELD_1_CONTAINED_MORE_THAN_THE_MAXIMUM_NUMBER_2_OF_CHARACTERS_ALLOWED
THE_OPTIONS_ARE_MUTUALLY_EXCLUSIVE_1_2_ARG2
TOO_MANY_RESULTS
TRANSLATION_HANDLE_IS_IN_AN_ILLEGAL_STATE
UNHANDLED_ERRORS_ON_RESTORE
UNKNOWN_CONTEXT_TYPE
UNKNOWN_PARAM
VALUE_1_OUTSIDE_OF_VALID_MINMAX_RANGE_FOR_FIELD_2
WORKBOOK_NAME_IS_MISSING
WRONG_PARAMETER_TYPE
WS_INVALID_REFERENCE_KEY_1
WS_NO_PERMISSIONS_TO_SET_VALUE
YOU_HAVE_ATTEMPTED_AN_UNSUPPORTED_ACTION

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myInvalidEmailsError = error.Type.INVALID_EMAILS_FOUND
4 ...
5 // Add additional code

```

N/file Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the file module when you want to work with files within NetSuite. You can use this module to upload files to the NetSuite File Cabinet. You can also use this module to send files as attachments without uploading them to the File Cabinet.

A [file.Reader](#) object, which is returned by [File.getReader\(\)](#), can be used for special read operations. Use [File.getSegments\(options\)](#) to retrieve an iterator of custom segments of a file.

Methods that load content in memory, such as [File.getContents\(\)](#), have a 10 MB size limit. This limit does not apply when content is streamed, such as when [File.save\(\)](#) is called.

- [N/file Module Members](#)
- [File Object Members](#)
- [N/file Module Script Samples](#)

N/file Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	file.File	object	Server-side scripts	Encapsulates a file within NetSuite.
	file.Reader	object	Server-side scripts	Encapsulates a reader that you can use to perform special read operations
Method	file.create(options)	file.File	Server-side scripts	Creates a new file.File .
	file.delete(options)	void	Server-side scripts	Deletes an existing file.File from the NetSuite File Cabinet.
	file.load(options)	file.File	Server-side scripts	Loads an existing file.File from the NetSuite File Cabinet.
Enum	file.Encoding	enum	Server-side scripts	Sets the value of the File.encoding property.
	file.Type	enum	Server-side scripts	Sets the value of the File.fileType property.

File Object Members

The following members are called on [file.File](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	File.appendLine(options)	<code>file.File</code> Object	Server-side scripts	Inserts a line to the end of a CSV or text file.
	File.getContents()	string	Server-side scripts	Returns the content of a file in string format.
	File.lines.iterator()	boolean true false	Server-side scripts	Calls a developer-defined function for each line. Returns false when line processing stops.
	File.resetStream()	void	Server-side scripts	Resets the file stream to its previous state.
	File.save()	number	Server-side scripts	Saves a new or updated file to the File Cabinet.
	File.getReader()	object	Server-side scripts	Returns reader object for read operations.
	File.getSegments(options)	object	Server-side scripts	Returns an iterator of segments that are delimited by the specified separator.
Property	File.description	string	Server-side scripts	Description of a file.
	File.encoding	string	Server-side scripts	Character encoding on a file.
	File.fileType	enum	Server-side scripts	File type of a file.
	File.folder	number	Server-side scripts	Internal ID of the folder that houses a file within the NetSuite File Cabinet.
	File.id	number (read-only)	Server-side scripts	Internal ID of a file in the NetSuite File Cabinet.
	File.isInactive	boolean true false	Server-side scripts	Inactive status of a file. If set to true, the file is inactive.
	File.isOnline	boolean true false	Server-side scripts	"Available without Login" status of a file. If set to true, users can download the file outside of a current NetSuite login session.
	File.isText	boolean (read-only)	Server-side scripts	Indicates whether a file type is text-based.
	File.name	string	Server-side scripts	Name of a file.
	File.path	string (read-only)	Server-side scripts	Relative path to a file in the NetSuite File Cabinet.
	File.size	number (read-only)	Server-side scripts	Size of a file in bytes.
	File.url	string (read-only)	Server-side scripts	URL of a file.

Reader Object Members

The following members are called on `file.Reader`.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Reader.readUntil(options)	string	Server-side scripts	Returns string from current position to the next occurrence of options.tag.
	Reader.readChars(options)	string	Server-side scripts	Returns the next options.number characters from the current position.

N/file Module Script Samples

The following script samples demonstrate how to use the features of the N/file module.

Sample 1: Create and save a file

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following script shows how to create and save a file to the File Cabinet. In this sample, the folder ID value is hard-coded. For the script to run in the SuiteScript Debugger, you must replace this hard-coded value with a valid folder ID from your account.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/file'], function(file) {
6     function createAndSaveFile() {
7         var fileObj = file.create({
8             name: 'test.txt',
9             fileType: file.Type.PLAINTEXT,
10            contents: 'Hello World\nHello World'
11        });
12        fileObj.folder = -15;
13
14        var id = fileObj.save();
15        fileObj = file.load({
16            id: id
17        });
18    }
19
20    createAndSaveFile();
21 });

```

Sample 2: Create and save a file using additional properties

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to create and save a file to the File Cabinet and also how to set the values of the [File.isOnLine](#) and the [File.folder](#) properties. In this sample, the folder ID value is hard-coded. For the script to run in the SuiteScript Debugger, you must replace this hard-coded value with a valid folder ID from your account.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/file'], function(file){
6      function createAndSaveFile(){
7          var fileObj = file.create({
8              name: 'test.txt',
9              fileType: file.Type.PLAINTEXT,
10             contents: 'Hello World\nHello World',
11             folder : -15,
12             isOnline : true
13         });
14
15         var id = fileObj.save();
16         fileObj = file.load({
17             id: id
18         });
19     }
20
21     createAndSaveFile();
22 });

```

Sample 3: Create a file and append lines

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a CSV file, appends several lines of data, and saves the file. The script also loads the file and calculates the total of several values in the file.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/file', 'N/error', 'N/log'], function (file, error, log) {
6     // This sample calculates the total for the
7     // second column value in a CSV file.
8     //
9     // Each line in the CSV file has the following format:
10    // date,amount
11    //
12    // Here is the data that the script adds to the file:
13    // 10/21/14,200.0
14    // 10/21/15,210.2
15    // 10/21/16,250.3
16
17    // Create the CSV file
18    var csvFile = file.create({
19        name: 'data.csv',
20        contents: 'date,amount\n',
21        folder: 39,
22        fileType: 'CSV'
23    });
24
25    // Add the data
26    csvFile.appendLine({
27        value: '10/21/14,200.0'
28    });
29    csvFile.appendLine({
30        value: '10/21/15,210.2'
31    });
32    csvFile.appendLine({
33        value: '10/21/16,250.3'
34    });

```

```

35 // Save the file
36 var csv fileId = csvFile.save();
37
38 // Create a variable to store the calculated total
39 var total = 0.0;
40
41 // Load the file
42 var invoiceFile = file.load({
43   id: csv fileId
44 });
45
46
47 // Obtain an iterator to process each line in the file
48 var iterator = invoiceFile.lines.iterator();
49
50 // Skip the first line, which is the CSV header line
51 iterator.each(function () {return false;});
52
53 // Process each line in the file
54 iterator.each(function (line) {
55   // Update the total based on the line value
56   var lineValues = line.value.split(',');
57   var lineAmount = parseFloat(lineValues[1]);
58   if (!lineAmount) {
59     throw error.create({
60       name: 'INVALID_INVOICE_FILE',
61       message: 'Invoice file contained non-numeric value for total: ' + lineValues[1]
62     });
63
64   total += lineAmount;
65   return true;
66 }
67 });
68
69 // At this point, the total is 660.5
70 log.debug({
71   title: 'total',
72   details: total
73 });
74 });

```

Sample 4: Implement a basic parser

i Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample reads and logs strings from a file using commas and new line characters as separators. This sample can be used as the starting point for a parser implementation.

```

1 /**
2  * @NApiVersion 2.0
3  * @NScriptType bankStatementParserPlugin
4 */
5
6 define(['N/file', 'N/log'], function(file, log) {
7   return {
8     parseBankStatement: function(context) {
9       var reader = context.input.file.getReader();
10
11       var textUntilFirstComma = reader.readUntil(',');
12       var next10Characters = reader.readChars(10);
13       var textUntilNextNewLine = reader.readUntil('\n');
14       var next100Characters = reader.readChars(100);
15
16       log.debug({

```

```

17     title: 'STATEMENT TEXT',
18     details: textUntilFirstComma
19   });
20
21   log.debug({
22     title: 'STATEMENT TEXT',
23     details: next10Characters
24   });
25
26   log.debug({
27     title: 'STATEMENT TEXT',
28     details: textUntilNextNewLine
29   });
30
31   log.debug({
32     title: 'STATEMENT TEXT',
33     details: next100Characters
34   })
35 }
36
37 });

```

Sample 5: Read a file in segments

i Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample reads and logs segments from a file using a set of characters as separators.

```

1 /**
2 * @NApiVersion 2.0
3 * @NScriptType bankStatementParserPlugin
4 */
5
6 define(['N/file', 'N/log'], function(file, log) {
7   return {
8     parseBankStatement: function(context) {
9       var statementFile = context.input.file;
10
11       var statementSegmentIterator = statementFile.getSegments({separator: '\\|_|/'}).iterator();
12       statementSegmentIterator.each(function (segment) {
13         log.debug({
14           title: 'STATEMENT TEXT',
15           details: segment.value
16         });
17         return true;
18       });
19     }
20   };
21 });

```

file.File

i Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a file within NetSuite.
--------------------	--------------------------------------

	<p>Note: This object only encapsulates a file's metadata. Content is only loaded into memory (and returned as a string) when you call the File.getContents(). Content from CSV or text files can be accessed line by line using File.appendLine(options) or File.lines.iterator().</p>
	<p>Important: Binary content must be base64 encoded.</p>
	<p>Create a new file.File Object (up to 10MB in size) with the file.create(options) method.</p> <p>After you create a new file.File, you can:</p> <ul style="list-style-type: none"> ■ upload it to the NetSuite File Cabinet with the File.save() method. ■ attach it to an email or fax without saving it to the File Cabinet.
	<p>Important: If you want to save the file to the NetSuite File Cabinet, you must set a NetSuite File Cabinet folder with the File.folder property. You must do this before you call File.save().</p>
	<p>Returns reader object File.getReader() and iterator of segments File.getSegments(options).</p> <p>For a complete list of this object's methods and properties, see File Object Members.</p>
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.create({
4   name: 'test.txt',
5   fileType: file.Type.PLAINTEXT,
6   contents: 'Hello World\nHello World'
7 });
8 fileObj.folder = 30;
9 var fileId = fileObj.save();
10 ...
11
12 //Add additional code

```

File.getContents()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to return the content of the file.
---------------------------	--

	 Important: Content held in memory is limited to 10MB.
	 Note: You can access CSV or text files (including files over 10MB) using <code>File.appendLine(options)</code> or <code>File.lines.iterator()</code> .
Returns	The file content as a string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/file Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_FILE_CONTENT_SIZE_EXCEEDED	The file content you are attempting to access exceeds the maximum allowed size of 10 MB.	You attempt to return the content of a file larger than 10MB.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 145
5 });
6 if (fileObj.size < 10485760){
7   fileObj.getContents();
8 }
9 ...
10 //Add additional code

```

File.getReader()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to return the reader object for performing special read operations
Returns	<code>file.Reader</code>
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/file Module
Since	2019.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 // Add additional code
2 ...
3 var reader = context.input.file.getReader();
4
5 var textUntilFirstComma = reader.readUntil(',');
6   var next10Characters = reader.readChars(10);
7   var textUntilNextNewLine = reader.readUntil('\n');
8   var next100Characters = reader.readChars(100);
9
10 ...
11 // Add additional code

```

File.getSegments(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to return the iterator of segments delimited by a separator. Separator is included in each segment. Empty separator is not allowed.
Returns	Iterator
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/file Module
Since	2019.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.separator	string	required	The separator to use to divide the segments. For example, if you specify a newline character as the separator, this method returns an iterator where each segment is a single line in the file.	2019.1

Errors

Error Code	Message	Thrown If
SSS_INVALID_SEGMENT_SEPARATOR	Segment separator must not be empty.	The options.separator argument is empty.
SSS_INVALID_ARG_TYPE	You have entered an invalid type argument: <passed type argument>	The options.separator argument is not a string.
SSS_MISSING_REQD_ARGUMENT	<name of missing parameter>	A required argument is not passed.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 // Add additional code
2 ...
3 var statementFile = context.input.file;
4
5 var statementSegmentIterator = statementFile.getSegments({
6   separator: '\\\\_|\\'
7 }).iterator();
8 statementSegmentIterator.each(function (segment) {
9
10 log.debug({
11   title: 'STATEMENT TEXT',
12   details: segment.value
13 });
14   return true;
15 ...
16 // Add additional code

```

File.appendLine(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to insert a line to the end of a file. This method can be used on text or .csv files.
	 Important: Content held in memory is limited to 10MB. Therefore, each line must be less than 10MB.
Returns	file.File Object
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/file Module
Since	2017.1

Parameters

i Note: The options parameter is a JavaScript object. The table below describes the name:value pairs that make up the object.

Parameter	Type	Required / Optional	Description	Since
options.value	string	required	Object containing a string to insert at the end of the file.	2017.1

Errors

Error Code	Message	Thrown If
SSS_FILE_CONTENT_SIZE_EXCEEDED	The content you are attempting to access exceeds the maximum allowed size of 10 MB.	You attempt to return the content of a line larger than 10MB.
YOU_CANNOT_WRITE_TO_A_FILE_AFTER_YOU_BEGAN_READING_FROM_IT		You call File.appendLine(options) after calling File.lines.iterator() . To avoid receiving the error, call File.resetStream() or save the file.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 145
5 });
6 fileObj.appendLine({
7   value: 'hello world'
8 });
9 ...
10 //Add additional code

```

File.lines.iterator()

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Method used to pass the next line as an argument to a developer-defined function. You can call this method multiple times to loop over the file contents as a stream.</p> <p>Return false to stop the loop. Return true to continue the loop. By default, false is returned when the end of the file is reached.</p> <p>This method can be used on text or .csv files.</p> <p>⚠ Important: Content held in memory is limited to 10MB. Therefore, each line must be less than 10MB.</p>
Returns	Boolean true false
Supported Script Types	<p>Server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>

Governance	None
Module	N/file Module
Since	2017.1

Parameters

Parameter	Type	Required / Optional	Description	Since
lineContext	iterator	required	Iterator which provides the next line of text from the text file to the iterator function.	2017.1

Errors

Error Code	Message	Thrown If
SSS_FILE_CONTENT_SIZE_EXCEEDED	The content you are attempting to access exceeds the maximum allowed size of 10 MB.	You attempt to return the content of a line larger than 10MB.
YOU_CANNOT_READ_FROM_A_FILE_AFTER_YOU_BEGAN_WRITING_TO_IT		You call File.lines.iterator() after calling File.appendLine(options) . Call File.resetStream() or save the file.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var iterator = invoiceFile.lines.iterator();
4
5 //Skip the first line (CSV header)
6 iterator.each(function () {return false;});
7 iterator.each(function (line)
8 {
9     // This function updates the total by
10    // adding the amount on each line to it
11    var lineValues = line.value.split(',');
12    var lineAmount = parseFloat(lineValues[1]);
13    if (!lineAmount)
14        throw error.create({
15            name: 'INVALID_INVOICE_FILE',
16            message: 'Invoice file contained non-numeric value for total: ' + lineValues[1]
17        });
18
19    total += lineAmount;
20    return true;
21 });
22 ...
23 //Add additional code

```

File.resetStream()

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to reset the file contents. Serves as an undo action on any unsaved content written with File.appendLine(options) or File.lines.iterator() .
---------------------------	--

	<p>Use this method to reset the reading and writing streams that may have been opened by <code>File.appendLine(options)</code> or <code>File.lines.iterator()</code>.</p> <p>The line pointer (or read iterator) is also set to its previous state.</p> <p>This method can be used on text or .csv files.</p>
	 Important: To use this method, each line must be less than 10MB.
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/file Module
Since	2017.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 //Add additional code
2 ...
3 var afile = file.create({
4   name: 'tmp3.txt',
5   fileType: 'PLAINTEXT',
6   contents:'one line'
7 });
8 afile.appendLine({
9   value:'line two'
10 });
11 afile.resetStream();
12 afile.lines(function f(){}
13 ...
14
15 //Add additional code

```

File.save()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to: <ul style="list-style-type: none"> ▪ Upload a new file to the NetSuite File Cabinet. ▪ Save an updated file to the NetSuite File Cabinet.
	 Note: The <code>File.save()</code> method streams files of any size, provided that the file to save or upload meets File Cabinet limits.
	 Important: If you want to save the file to the NetSuite File Cabinet, you must set a NetSuite File Cabinet folder with the <code>File.folder</code> property. You must do this before you call <code>File.save()</code> .
Returns	The internal ID of the file as a number.

Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/file Module
Since	2015.2

Errors

Error Code	Message	Thrown If
INVALID_KEY_OR_REF	Invalid folder reference key <passed folder ID>.	The File.folder property is set to an invalid folder ID.
SSS_MISSING_REQD_ARGUMENT	Please enter value(s) for: Folder	The File.folder property is not set before save() is called.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.create({
4     name : 'test.txt',
5     fileType: file.Type.PLAINTEXT,
6     contents: 'Hello World\nHello World'
7 });
8 fileObj.folder = 30;
9 var fileId = fileObj.save();
10 ...
11 //Add additional code

```

File.description



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The description of a file. In the UI, the value of description displays in the Description field on the file record.
Type	string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 'Images/myImageFile.jpg'
5 });
6 fileObj.description = 'my test file';
7 var fileId = fileObj.save();
8 ...
9 //Add additional code

```

File.encoding



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The character encoding on a file. Value is set with the file.Encoding enum.
Type	string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 //Add additional code
2 ...
3 var fileObj = file.create({
4   name : 'test.txt',
5   fileType: file.Type.PLAINTEXT,
6   contents: 'Hello World\nHello World'
7 });
8 fileObj.encoding = file.Encoding.MAC_ROMAN;
9 fileObj.folder = 30;
10 var fileId = fileObj.save();
11 ...
12
13 //Add additional code

```

File.fileType



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The file type of a file. This property is read-only. You must set the file type by passing in a file.Type enum value to file.create(options) .
-----------------------------	---

Type	enum
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY_PROPERTY		You attempt to edit this property after it is set with file.create(options) .

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 145
5 });
6 log.debug({
7   details: "File Type: " + fileObj.fileType
8 });
9 ...
10
11 //Add additional code

```

File.folder

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of a file's folder within the NetSuite File Cabinet. Before you upload a file to the NetSuite File Cabinet with File.save() , you must set its File Cabinet folder with the folder property.
Type	number string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.create({

```

```

4   name: 'test.txt',
5   fileType: file.Type.PLAINTEXT,
6   contents: 'Hello World\nHello World'
7 });
8 fileObj.folder = 30;
9 var fileId = fileObj.save();
10 ...
11 //Add additional code

```

File.id

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of the file within the NetSuite File Cabinet. This value is automatically generated by NetSuite. This property is read-only.
Type	number
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY_PROPERTY		You attempt to edit this property.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 'Images/myImageFile.jpg'
5 });
6 log.debug({
7   details: "File ID: " + fileObj.id
8 });
9 ...
10 //Add additional code

```

File.isInactive

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The inactive status of a file. If set to true, the file is inactive. The default value is false. When a file is inactive, it does not display in the UI unless you select Show Inactives on the File Cabinet page.
-----------------------------	---

Type	boolean true false
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 'Images/myImageFile.jpg'
5 });
6 fileObj.name = 'myOldImageFile.jpg';
7 fileObj.isInactive = true;
8 var fileId = fileObj.save();
9 ...
10 //Add additional code

```

File.isOnLine

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The Available without Login status of a file. If set to true, users can download the file outside of a current NetSuite login session. The default value is false.  Important: This property holds the value of the Available without Login setting found on the file record. It does not reflect the value of the Available Without Login setting found on the Suitelet script deployment record. The Available without Login setting is primarily used for SuiteCommerce websites. When this setting is enabled, websites can access media files in the NetSuite File Cabinet without a current NetSuite login session.
Type	boolean true false
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```
1 //Add additional code
```

```

2 ...
3 var fileObj = file.load({
4   id: 'Images/myImageFile.jpg'
5 });
6 fileObj.isOnline = true;
7 var fileId = fileObj.save();
8 ...
9 //Add additional code

```

File.isText

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether a file type is text-based. This property is read-only.
Type	boolean true false
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY_PROPERTY		You attempt to edit this property.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 145
5 });
6 if (fileObj.isText === true){
7   ...
8 }
9 ...
10 //Add additional code

```

File.name

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of a file.
Type	string
Supported Script Types	Server-side scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 'Images/myImageFile.jpg'
5 });
6 fileObj.name = 'myOldImageFile.jpg';
7 var fileId = fileObj.save();
8 ...
9 //Add additional code

```

File.path

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The relative path to a file in the NetSuite File Cabinet.
	<p>Note: If the folder is not set with the file.create(options) method, this property holds the file name until the File.folder property is defined.</p>
	This property is read-only.
Type	string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY_PROPERTY		You attempt to edit this property.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 145
5 });
6 log.debug({

```

```

7   details: "File Path: " + fileObj.path
8 });
9 ...
10 //Add additional code

```

File.size



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The size of a file in bytes. This property is read-only.
Type	number
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY_PROPERTY		You attempt to edit this property.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.load({
4   id: 'Images/myImageFile.jpg'
5 });
6 log.debug({
7   details: "File Size: " + fileObj.size
8 });
9 ...
10 //Add additional code

```

File.url



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The URL of a file. This property is read-only.
Type	string

Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY_PROPERTY		You attempt to edit this property.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code ...
2 var fileObj = file.load({
3   id: 'Images/myImageFile.jpg'
4 });
5 log.debug({
6   details: "File URL: " + fileObj.url
7 });
8 ...
9 //Add additional code

```

file.create(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to create a new file in the NetSuite File Cabinet.  Important: Content held in memory is limited to 10MB.
Returns	file.File
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/file Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	■ The file name and extension.	2015.2

Parameter	Type	Required / Optional	Description	Since
			<ul style="list-style-type: none"> ■ Sets the value for the File.name property. 	
options.fileType	enum	required	<ul style="list-style-type: none"> ■ The file type. ■ Sets the value for the File.fileType property. This property is read-only and cannot be changed after the file is created. ■ Use the file.Type enum to set the value. 	2015.2
options.contents	string	optional	<ul style="list-style-type: none"> ■ The file content. ■ File content is lazy loaded; there is no property for it. ■ If the file type is binary (for example, PDF), the file content must be base64 encoded. 	2015.2
options.description	string	optional	<ul style="list-style-type: none"> ■ The file description. In the UI, the value of description displays the Description field on the file record. ■ Sets the value for the File.description property. 	2016.2
options.folder	number	optional	<ul style="list-style-type: none"> ■ The internal ID of the folder within the NetSuite File Cabinet. You must set the File Cabinet folder before you upload a file to the NetSuite File Cabinet with File.save(). ■ Sets the value for the File.folder property. 	2016.1
options.encoding	string	optional	<ul style="list-style-type: none"> ■ The character encoding on a file. ■ Sets the value for the File.encoding property. ■ Use the file.Encoding enum to set the value. 	2016.2
options.isInactive	boolean false true	optional	<ul style="list-style-type: none"> ■ The inactive status of a file. If set to true, the file is inactive. The default value is false. When a file is inactive, it does not display in the UI unless you select Show Inactives on the File Cabinet page. ■ Sets the value for the File.isInactive property. 	2016.2
options.isOnLine	boolean false true	optional	<ul style="list-style-type: none"> ■ The Available without Login status of a file. If set to true, users can download the file outside of a current netSuite login session. The default value is false. ■ Sets the value for the File.isOnLine property. 	2016.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	<name of missing parameter>	A required argument is not passed.
SSS_INVALID_TYPE_ARG	You have entered an invalid type argument: <passed type argument>	The argument for File.fileType is invalid.

Error Code	Message	Thrown If
SSS_FILE_CONTENT_SIZE_EXCEEDED	The file you are trying to create exceeds the maximum allowed file size of 10.0 MB.	You attempt to create a file larger than 10MB.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var fileObj = file.create({
4   name: 'test.txt',
5   fileType: file.Type.PLAINTEXT,
6   contents: 'Hello World\nHello World',
7   description: 'This is a plain text file.',
8   encoding: file.Encoding.UTF8,
9   folder: 30,
10  isOnline: true
11 });
12 ...
13 //Add additional code

```

file.delete(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to delete an existing file from the NetSuite File Cabinet.
Returns	The internal ID of the deleted file
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/file Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	number string	required	<ul style="list-style-type: none"> ■ Internal ID of the file. ■ To find the internal ID of the file in the UI, click Documents > Files > File Cabinet. 	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	<name of missing parameter>	A required argument is not passed.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 //Add additional code
2 ...
3 var fileObj = file.create({
4   name: 'test.txt',
5   fileType: file.Type.PLAINTEXT,
6   contents: 'Hello World\nHello World'
7 });
8 fileObj.folder = 30;
9 var fileId = fileObj.save();
10
11 file.delete({
12   id: fileId
13 });
14 ...
15 //Add additional code

```

file.load(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Loads an existing file from the NetSuite File Cabinet. The file size limit for this method is 2GB.
Returns	file.File
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/file Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	number string	required	The identifier of a file in the File Cabinet. To specify a file in the File Cabinet, you can pass one of the following as the value of this parameter: <ul style="list-style-type: none">■ The internal ID of the file as a number or string	2015.2

Parameter	Type	Required / Optional	Description	Since
			<ul style="list-style-type: none"> ■ The absolute file path to the file in the File Cabinet (for example, 'Images/myImageFile.jpg') ■ The relative file path to the file in the File Cabinet (for example, './Images/myImageFile.jpg' to specify a file path relative to the current folder of your script, or '../Images/myImageFile.jpg' to specify a file path relative to the parent folder of your script) <p>To find the internal ID of the file in the UI, select Documents > Files > File Cabinet.</p>	

Errors

Error Code	Message	Thrown If
INSUFFICIENT_PERMISSION	You do not have access to the media item you selected.	Internal ID passed is invalid.
RCRD_DSNT_EXIST	That record does not exist. path: {path}	Relative file path passed is invalid.
SSS_MISSING_REQD_ARGUMENT	<name of missing parameter>	A required argument is not passed.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [file Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var fileObj = file.load({
4   id: 'Images/myImageFile.jpg'
5 });
6 fileObj.description = 'my test file';
7 var fileId = fileObj.save();
8 ...
9 // Add additional code

```

file.Encoding

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Enumeration that holds the string values for supported character encoding.</p> <p>This enum is used to set the value of the File.encoding property.</p> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	<p>Server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/file Module

Since	2015.2
-------	--------

Values

Value	Character Set
UTF_8	Unicode
WINDOWS_1252	Western
ISO_8859_1	Western
GB18030	Chinese Simplified
SHIFT_JIS	Japanese
MAC_ROMAN	Western
GB2312	Chinese Simplified
BIG5	Chinese Traditional

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 //Add additional code
2 ...
3 var fileObj = file.create({
4   name: 'test.txt',
5   fileType: file.Type.PLAINTEXT,
6   contents: 'Hello World\nHello World'
7 });
8 fileObj.encoding = file.Encoding.MAC_ROMAN;
9 fileObj.folder = 30;
10 var fileId = fileObj.save();
11 ...
12 //Add additional code

```

file.Type

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Enumeration that holds the string values for supported file types. This enum is used to set the value of the File.fileType property.</p> <p>Note that the File.fileType property is read only. Its value must be set with file.create(options). See file Module Code Sample for an example.</p> <p> Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	<p>Server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>

Module	N/file Module
Since	2015.2

Values

■ APPCACHE	■ JSON	■ RTF
■ AUTOCAD	■ MESSAGERFC	■ SCSS
■ BMPIMAGE	■ MP3	■ SMS
■ CERTIFICATE	■ MPEGMOVIE	■ STYLESHEET
■ CONFIG	■ MSPROJECT	■ SVG
■ CSV	■ PDF	■ TAR
■ EXCEL	■ PJPGIMAGE	■ TIFFIMAGE
■ FLASH	■ PLAINTEXT	■ VISIO
■ FREEMARKER	■ PNGIMAGE	■ WEBAPPPAGE
■ GIFIMAGE	■ POSTSCRIPT	■ WEBAPPSCRIPT
■ GZIP	■ POWERPOINT	■ WORD
■ HTMLDOC	■ QUICKTIME	■ XMLDOC
■ ICON		■ XSD
■ JAVASCRIPT		■ ZIP
■ JPGIMAGE		

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 //Add additional code
2 ...
3 var fileObj = file.create({
4   name : 'test.txt',
5   fileType: file.Type.PLAINTEXT,
6   contents: 'Hello World\nHello World'
7 });
8 fileObj.folder = 30;
9 var fileId = fileObj.save();
10 ...
11 //Add additional code

```

file.Reader



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Use for special read operations. Reads from a file until a specified delimiter is reached. Reads an arbitrary number of characters from a file.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/file Module

Methods and Properties	Reader Object Members
Since	2019.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 var reader = context.input.file.getReader();
2
3     var textUntilFirstComma = reader.readUntil(',');
4     var next10Characters = reader.readChars(10);
5     var textUntilNextNewLine = reader.readUntil('\n');
6     var next100Characters = reader.readChars(100);

```

Reader.readUntil(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns string from current position to the next occurrence of options.tag. Returns the rest of the string if tag is not found. Returns null if reading is already finished. All types of characters are supported. If there's a character that does not exist until the end of the file, the rest of the file is returned.
Returns	string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/file Module
Since	2019.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.tag	string	required	String containing a tag	2019.1

Errors

Error Code	Message	Thrown If
SSS_TAG_CANNOT_BE_EMPTY	Tag cannot be empty.	The options.tag argument is empty.
SSS_INVALID_ARG_TYPE	You have entered an invalid type argument: <passed type argument>	The options.tag argument is not a string.

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	<name of missing parameter>	A required argument is not passed.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 // Add additional code
2 ...
3 var reader = context.input.file.getReader();
4
5 var textUntilFirstComma = reader.readUntil(',');
6 var next10Characters = reader.readChars(10);
7 var textUntilNextNewLine = reader.readUntil('\n');
8 var next100Characters = reader.readChars(100);
9
10 ...
11 // Add additional code

```

Reader.readChars(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the next options.number characters from the current position. Returns less than the number if there is not enough characters to read in the file. Returns null if reading is already finished.
Returns	string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/file Module
Since	2019.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.number	number	required	The number of characters to read.	2019.1

Errors

Error Code	Message	Thrown If
SSS_INVALID_READ_SIZE	Read size must be positive.	The options.number argument is not greater than zero.

Error Code	Message	Thrown If
SSS_INVALID_ARG_TYPE	You have entered an invalid type argument: <passed type argument>	The options.number argument is not a number.
SSS_MISSING_REQD_ARGUMENT	<name of missing parameter>	A required argument is not passed.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see file Module Script Sample.

```

1 // Add additional code
2 ...
3 var reader = context.input.file.getReader();
4
5 var textUntilFirstComma = reader.readUntil(',');
6 var next10Characters = reader.readChars(10);
7 var textUntilNextNewLine = reader.readUntil('\n');
8 var next100Characters = reader.readChars(100);
9
10 ...
11 // Add additional code

```

N/format Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Use the format module to parse formatted data into strings and to convert strings into a specified format. The format module formats data according to personal preferences set on the Set Preferences page, accessible from Home > Set Preferences . See the help topic [Setting Personal Preferences](#).

- [N/format Module Members](#)
- [N/format Module Script Samples](#)

N/format Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	format.format(options)	string Date	Client and server-side scripts	Takes a raw value and returns a formatted value. <div style="border: 1px solid #0070C0; padding: 5px; background-color: #E0F2FD;"> i Note: This method is overloaded when you format a datetime or datetimetz value. </div>
	format.parse(options)	Date string number	Client and server-side scripts	Takes a formatted value and returns a raw value.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				<p>Note: This method is overloaded when you format a datetime or datetimetz value.</p>
Enum	format.Type	enum	Client and server-side scripts	Holds the string values for the supported field types. This enum is used to set the value of the options.type parameter.
	format.Timezone	enum	Client and server-side scripts	Holds the string values for supported time zone formats. This enum is used to set the value of the options.timezone parameter.

N/format Module Script Samples

For help with writing scripts in SuiteScript 2.0, see the help topics [SuiteScript 2.0 Hello World](#) and [SuiteScript 2.0 Entry Point Script Creation and Deployment](#).

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample parses a string (formatted according to the user preferences) to a raw Date Object, and then parses it back to the formatted string. This sample uses `format.parse(options)` and `format.format(options)`.

This sample assumes the Date Format set in the preferences is MM/DD/YYYY. You may need to change the value for the date used in this script to match the preferences set in your account.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4 */
5 define(['N/ui/serverWidget', 'N/format'], function(serverWidget, format) {
6     function parseAndFormatDateString() {
7         // Assuming Date format is MM/DD/YYYY
8         var initialFormattedDateString = "07/28/2015";
9         var parsedDateStringAsRawDateObject = format.parse({
10             value: initialFormattedDateString,
11             type: format.Type.DATE
12         });
13         var formattedDateString = format.format({
14             value: parsedDateStringAsRawDateObject,
15             type: format.Type.DATE
16         });
17         return [parsedDateStringAsRawDateObject, formattedDateString];
18     }
19     function onRequest(context) {
20         var data = parseAndFormatDateString();
21
22         var form = serverWidget.createForm({
23             title: "Date"
24         });

```

```

25     var fldDate = form.addField({
26       type: serverWidget.FieldType.DATE,
27       id: "date",
28       label: "Date"
29     });
30     fldDate.defaultValue = data[0];
31
32     var fldString = form.addField({
33       type: serverWidget.FieldType.TEXT,
34       id: "dateastext",
35       label: "Date as text"
36     });
37     fldString.defaultValue = data[1];
38
39     context.response.writePage(form);
40   }
41   return {
42     onRequest: onRequest
43   };
44 });
45 });

```

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample parses a string (formatted according to the user preference) to a raw number value, using `format.parse(options)`.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/format'],
6   function(format){
7     function parseToValue() {
8       // Assume number format is 1.000,00,00 and negative format is -100
9       var formattedNum = "-20.000,25"
10      return format.parse({value:formattedNum, type: format.Type.FLOAT})
11    }
12    var rawNum = parseToValue(); // -20000.25 -- a number
13  });

```

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample formats a raw number value (formatted according to the user preference) to a string, using `format.format(options)`.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/format'],
6   function(format){
7     function formatToString() {
8       // Assume number format is 1.000,00,00 and negative format is (100)
9       var rawNum2 = -4444.44
10      return format.format({value:rawNum2, type: format.Type.FLOAT})
11    }
12    var formattedNum2 = formatToString(); // "44.444,44" -- a string

```

```
13 |});
```

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample formats the time of day to a string, using `format.format(options)`.

```
1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/format'],
6   function(format){
7     function formatTimeOfDay() {
8       // Assume the time format is hh:mm (24 hours)
9       var now = new Date(); // Say it's 7:01PM right now.
10      return format.format({value: now, type: format.Type.TIMEOFDAY})
11    }
12    var formattedTime = formatTimeOfDay(); // "19:01" – a string
13  });

```

format.format(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Formats a value from the raw value to its appropriate preference format. i Note: This method is overloaded when you format a datetime or datetimetz value.
Returns	If a datetime or datetimetz value is specified, the string in date format is returned in the user's local app time zone. i Note: If an invalid value is given, the original value passed to options.value is returned. i Note: For client side scripts, the string returned is based on the user's system time. For server-side scripts, the string returned is based on the system time of the server your NetSuite system is running on.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types
Governance	None
Module	N/format Module
Since	2015.2

Parameters

This method is overloaded when you format a datetime or datetimetz value.



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.value	Date string number	required	The input data to format.	2015.2
options.type	string	required	The field type (for example, DATE, CURRENCY, INTEGER). Set using the format.Type enum.	2015.2

The table below applies to datetime and datetimetz values only.

Parameter	Type	Required / Optional	Description	Since
options.value	Date	required	The Date Object being converted into a string	2015.2
options.type	string	required	The field type (either DATETIME or DATETIMETZ). Set using the format.Type enum.	2015.2
options.timezone	enum number	optional	The time zone specified for the returned string. Set using the format.Timezone enum or key. If a time zone is not specified, the time zone is set based on user preference. If the time zone is invalid, the time zone is set to GMT.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format Module Script Samples](#).

```

1 // Add additional code
2 ...
3     function(format){
4         function formatToString() {
5             // Assume number format is 1.000.000,00 and negative format is (100)
6             var rawNum2 = -44444.44
7             return format.format({value:rawNum2, type: format.Type.FLOAT})
8         }
9         var formattedNum2 = formatToString(); // "44.444,44" -- a string
10 ...
11 // Add additional code

```

format.parse(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Parses a value from the appropriate preference format to its raw value. The appropriate preference format is the one selected in the Date Format field at Home > Set Preferences.
---------------------------	---

	For a datetime or datetimetz value, use this method to convert a Date Object into a string based on the specified timezone.
	<p>Note: This method is overloaded when you format a datetime or datetimetz value.</p>
Returns	Datetime or datetimetz values are returned as a Date Object.
	<p>Note: If the value given is not valid or parseable, the original value passed to options.value is returned.</p>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types
Governance	None
Module	N/format Module
Since	2015.2

Parameters

This method is overloaded when you format a datetime or datetimetz value.

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.value	string	required	The input data to parse.	2015.2
options.type	string	required	The field type (for example, DATE, CURRENCY, INTEGER). Set using the format.Type enum.	2015.2

The table below applies to datetime and datetimeTZ values only.

Parameter	Type	Required / Optional	Description	Since
options.value	string	required	The string that contains the date and time information in the specified timezone.	2015.2
options.type	string	required	The field type (either DATETIME or DATETIMETZ). Set using the format.Type enum.	2015.2
options.timezone	enum	optional	The time zone represented by the options.value string. Set using the format.Timezone enum. If a time zone is not specified, the time zone is based on user preference. If the time zone is invalid, the time zone is set to GMT.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format Module Script Samples](#).

```

1 // Add additional code
2 ...
3     function(format){
4         function parseToValue() {
5             // Assume number format is 1.000.000,00 and negative format is -100
6             var formattedNum = "-20.000,25"
7             return format.parse({value:formattedNum, type: format.Type.FLOAT})
8         }
9         var rawNum = parseToValue(); // -20000.25 -- a number
10 ...
11 // Add additional code

```

format.Type



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Enumeration that holds the string values for the supported field types. This enum is used to set the value of the options .type parameter when calling format.format(options) or format.parse(options) .
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types
Module	N/format Module
Since	2015.2

Values

■ ADDRESS	■ EMAIL	■ POSCURRENCY
■ CCEXPDATE	■ EMAILS	■ POSFLOAT
■ CCNUMBER	■ FLOAT	■ POSINTEGER
■ CCVALIDFROM	■ FULLPHONE	■ QUOTEDFUNCTION
■ CHECKBOX	■ FUNCTION	■ RADIO
■ COLOR	■ FURIGANA	■ RATE
■ CURRENCY	■ IDENTIFIER	■ RATEHIGHPRECISION
■ CURRENCY2	■ IDENTIFIERANYCASE	■ SELECT
■ DATE	■ INTEGER	■ TEXT
■ DATETIME	■ MMYYDATE	■ TEXTAREA
■ DATETIMETZ	■ NONNEGURRENCY	■ TIME
■ DOCUMENT	■ NONNEGFLOAT	■ TIMEOFDAY
■ DYNAMICPRECISION	■ PACKAGE	■ TIMETRACK
	■ PERCENT	

	■ PHONE	■ URL
--	---------	-------

Be aware of the following:

- The following field types require a value of greater than 0:
 - POSCURRENCY
 - POSINTEGER
 - POSINTEGER
- NONNEGFLOAT requires a value that is greater than or equal to 0
- CURRENCY field type rounds the number based on the user's currency precision setting and is limited to hundredths / 2 decimals (0.00).
- CURRENCY2 field type formats using a record's currency precision.
- If any of the following field types is set to hidden, the object returned is text.
 - Checkbox
 - Radio
 - Select
 - Textarea
- DYNAMICPRECISION is controlled by NetSuite and can differ from field to field.

Syntax



Note: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format Module Script Samples](#).

```

1 //Run this snippet in debugger to see how values are formatted.
2 ...
3 ...
4 require(['N/format'], function(format) {
5   Object.getOwnPropertyNames(format.Type).forEach(function(type) {
6     log.debug(type, format.format({
7       value: 12345,
8       type: format.Type[type]
9     }));
10    /*use any value instead of 12345 to see how it is formatted*/
11  })
12 })
```

```

1 // Add additional code
2 ...
3 function formatTimeOfDay() {
4   // Assume the time format is hh:mm (24 hours)
5   var now = new Date(); // Say it's 7:01PM right now.
6   var formattedTime = format.format({value: now, type: format.Type.TIMEOFDAY})
7 }
8 ...
9 ...
10 // Add additional code
```

```

1 require(['N/format'],
2   function(format) {
3 ...
4   function parseAndFormatDateString()
5   {
6     var rawDateString = "07/28/2015";
7     var parsedDate= format.parse(
8       {value: rawDateString,
9        type: format.Type.DATE
10 });
11    console.log(parsedDate);
12 })
```

```

12 }
13     parseAndFormatDateString();
14 }
15 );
16 );

```

format.Timezone

- Note:** The content in this help topic pertains to SuiteScript 2.0.
- Note:** JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Enumeration that holds the string values for supported time zone formats. This enum is used to set the value of the options.timezone parameter when calling format.format(options) or format.parse(options) .
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types
Module	N/format Module
Since	2015.2

Values

This table defines all valid time zone names in Olson Value format and includes daylight savings time rules for each time zone. Olson Values are maintained by the International Assigned Numbers Authority (IANA) in an international standard time zone database. The values that populate the Time Zone dropdown list found at Home > Set Preferences are also based on these values.

When working with alternate time zones in SuiteScript, use these enumeration values. If necessary, you can use the numerical key in place of an Olson Value string. For example, to source a custom timezone dropdown list.

Key	Olson Value	Description
1	ETC_GMT_PLUS_12: 'Etc/GMT+12'	(GMT-12:00) International Date Line West
2	PACIFIC_SAMOA: 'Pacific/Samoa'	(GMT-11:00) Midway Island, Samoa
3	PACIFIC_HONOLULU: 'Pacific/Honolulu'	(GMT-10:00) Hawaii
4	AMERICA_ANCHORAGE: 'America/Anchorage'	(GMT-09:00) Alaska
5	AMERICA_LOS_ANGELES: 'America/Los_Angeles'	(GMT-08:00) Pacific Time (US & Canada)
6	AMERICA_TIJUANA: 'America/Tijuana'	(GMT-08:00) Tijuana, Baja California
7	AMERICA_DENVER: 'America/Denver'	(GMT-07:00) Mountain Time (US & Canada)
8	AMERICA_PHOENIX: 'America/Phoenix'	(GMT-07:00) Arizona
9	AMERICA_CHIHUAHUA: 'America/Chihuahua'	(GMT-07:00) Chihuahua, La Paz, Mazatlan - New
10	AMERICA_CHICAGO: 'America/Chicago'	(GMT-06:00) Central Time (US & Canada)
11	AMERICA_REGINA: 'America/Regina'	(GMT-06:00) Saskatchewan

Key	Olson Value	Description
12	AMERICA_GUATEMALA: 'America/Guatemala'	(GMT-06:00) Central America
13	AMERICA_MEXICO_CITY: 'America/Mexico_City'	(GMT-06:00) Guadalajara, Mexico City, Monterrey - Old
14	AMERICA_NEW_YORK: 'America/New_York'	(GMT-05:00) Eastern Time (US & Canada)
15	US_EAST_INDIANA: 'US/East-Indiana'	(GMT-05:00) Indiana (East)
16	AMERICA_BOGOTA: 'America/Bogota'	(GMT-05:00) Bogota, Lima, Quito
17	AMERICA_CARACAS: 'America/Caracas'	(GMT-04:30) Caracas
18	AMERICA_HALIFAX: 'America/Halifax'	(GMT-04:00) Atlantic Time (Canada)
19	AMERICA_LA_PAZ: 'America/La_Paz'	(GMT-04:00) Georgetown, La Paz, San Juan
20	AMERICA_MANAUS: 'America/Manaus'	(GMT-04:00) Manaus
21	AMERICA_SANTIAGO: 'America/Santiago'	(GMT-04:00) Santiago
22	AMERICA_ST_JOHNS: 'America/St_Johns'	(GMT-03:30) Newfoundland
23	AMERICA_SAO_PAULO: 'America/Sao_Paulo'	(GMT-03:00) Brasilia
24	AMERICA_BUENOS_AIRES: 'America/Buenos_Aires'	(GMT-03:00) Buenos Aires
25	ETC_GMT_PLUS_3: 'Etc/GMT+3'	(GMT-03:00) Cayenne
26	AMERICA_GODTHAB: 'America/Godthab'	(GMT-03:00) Greenland
27	AMERICA_MONTEVIDEO: 'America/Montevideo'	(GMT-03:00) Montevideo
28	AMERICA_NORONHA: 'America/Noronha'	(GMT-02:00) Mid-Atlantic
29	ETC_GMT_PLUS_1: 'Etc/GMT+1'	(GMT-01:00) Cape Verde Is.
30	ATLANTIC_AZORES: 'Atlantic/Azores'	(GMT-01:00) Azores
31	EUROPE_LONDON: 'Europe/London', GMT: 'GMT'	(GMT) Greenwich Mean Time : Dublin, Edinburgh, Lisbon, London
32	GMT: 'GMT'	(GMT) Casablanca
33	ATLANTIC_REYKJAVIK: 'Atlantic/Reykjavik'	(GMT) Monrovia, Reykjavik
34	EUROPE_WARSAW: 'Europe/Warsaw'	(GMT+01:00) Sarajevo, Skopje, Warsaw, Zagreb
35	EUROPE_PARIS: 'Europe/Paris'	(GMT+01:00) Brussels, Copenhagen, Madrid, Paris
36	ETC_GMT_MINUS_1: 'Etc/GMT-1'	(GMT+01:00) West Central Africa
37	EUROPE_AMSTERDAM: 'Europe/Amsterdam'	(GMT+01:00) Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna
38	EUROPE_BUDAPEST: 'Europe/Budapest'	(GMT+01:00) Belgrade, Bratislava, Budapest, Ljubljana, Prague
39	AFRICA_CAIRO: 'Africa/Cairo'	(GMT+02:00) Cairo
40	EUROPE_ISTANBUL: 'Europe/Istanbul'	(GMT+02:00) Athens, Bucharest, Istanbul
41	ASIA_JERUSALEM: 'Asia/Jerusalem'	(GMT+02:00) Jerusalem

Key	Olson Value	Description
42	ASIA_AMMAN: 'Asia/Amman'	(GMT+02:00) Amman
43	ASIA_BEIRUT: 'Asia/Beirut'	(GMT+02:00) Beirut
44	AFRICA_JOHANNESBURG: 'Africa/Johannesburg'	(GMT+02:00) Harare, Pretoria
45	EUROPE_KIEV: 'Europe/Kiev'	(GMT+02:00) Helsinki, Kyiv, Riga, Sofia, Tallinn, Vilnius
46	EUROPE_MINSK: 'Europe/Minsk'	(GMT+02:00) Minsk
47	AFRICA_WINDHOEK: 'Africa/Windhoek'	(GMT+02:00) Windhoek
48	ASIA_RIYADH: 'Asia/Riyadh'	(GMT+03:00) Kuwait, Riyadh
49	EUROPE_MOSCOW: 'Europe/Moscow'	(GMT+03:00) Moscow, St. Petersburg, Volgograd
50	ASIA_BAGHDAD: 'Asia/Baghdad'	(GMT+03:00) Baghdad
51	AFRICA_NAIROBI: 'Africa/Nairobi'	(GMT+03:00) Nairobi
52	ASIA_TEHRAN: 'Asia/Tehran'	(GMT+03:30) Tehran
53	ASIA_MUSCAT: 'Asia/Muscat'	(GMT+04:00) Abu Dhabi, Muscat
54	ASIA_BAKU: 'Asia/Baku'	(GMT+04:00) Baku
55	ASIA_YEREVAN: 'Asia/Yerevan'	(GMT+04:00) Caucasus Standard Time
56	ETC_GMT_MINUS_3: 'Etc/GMT-3'	(GMT+04:00) Tbilisi
57	ASIA_KABUL: 'Asia/Kabul'	(GMT+04:30) Kabul
58	ASIA_KARACHI: 'Asia/Karachi'	(GMT+05:00) Islamabad, Karachi
59	ASIA_YEKATERINBURG: 'Asia/Yekaterinburg'	(GMT+05:00) Ekaterinburg
60	ASIA_TASHKENT: 'Asia/Tashkent'	(GMT+05:00) Tashkent
61	ASIA_CALCUTTA: 'Asia/Calcutta'	(GMT+05:30) Chennai, Kolkata, Mumbai, New Delhi
62	ASIA_KATMANDU: 'Asia/Katmandu'	(GMT+05:45) Kathmandu
63	ASIA_ALMATY: 'Asia/Almaty'	(GMT+06:00) Novosibirsk
64	ASIA_DHAKA: 'Asia/Dhaka'	(GMT+06:00) Astana, Dhaka
65	ASIA_RANGOON: 'Asia/Rangoon'	(GMT+06:30) Yangon (Rangoon)
66	ASIA_BANGKOK: 'Asia/Bangkok'	(GMT+07:00) Bangkok, Hanoi, Jakarta
67	ASIA_KRASNOYARSK: 'Asia/Krasnoyarsk'	(GMT+07:00) Krasnoyarsk
68	ASIA_HONG_KONG: 'Asia/Hong_Kong'	(GMT+08:00) Beijing, Chongqing, Hong Kong, Urumqi
69	ASIA_KUALA_LUMPUR: 'Asia/Kuala_Lumpur'	(GMT+08:00) Kuala Lumpur, Singapore
70	ASIA_TAIPEI: 'Asia/Taipei'	(GMT+08:00) Taipei
71	AUSTRALIA_PERTH: 'Australia/Perth'	(GMT+08:00) Perth
72	ASIA_IRKUTSK: 'Asia/Irkutsk'	(GMT+08:00) Irkutsk
73	ASIA_MANILA: 'Asia/Manila'	(GMT+08:00) Manila

Key	Olson Value	Description
74	ASIA_SEOUL: 'Asia/Seoul'	(GMT+09:00) Seoul
75	ASIA_TOKYO: 'Asia/Tokyo'	(GMT+09:00) Osaka, Sapporo, Tokyo
76	ASIA_YAKUTSK: 'Asia/Yakutsk'	(GMT+09:00) Yakutsk
77	AUSTRALIA_DARWIN: 'Australia/Darwin'	(GMT+09:30) Darwin
78	AUSTRALIA_ADELAIDE: 'Australia/Adelaide'	(GMT+09:30) Adelaide
79	AUSTRALIA_SYDNEY: 'Australia/Sydney'	(GMT+10:00) Canberra, Melbourne, Sydney
80	AUSTRALIA_BRISBANE: 'Australia/Brisbane'	(GMT+10:00) Brisbane
81	AUSTRALIA_HOBART: 'Australia/Hobart'	(GMT+10:00) Hobart
82	PACIFIC_GUAM: 'Pacific/Guam'	(GMT+10:00) Guam, Port Moresby
83	ASIA_VLADIVOSTOK: 'Asia/Vladivostok'	(GMT+10:00) Vladivostok
84	ASIA_MAGADAN: 'Asia/Magadan'	(GMT+11:00) Magadan, Solomon Is., New Caledonia
85	PACIFIC_KWAJALEIN: 'Pacific/Kwajalein'	(GMT+12:00) Fiji, Marshall Is.
86	PACIFIC_AUCKLAND: 'Pacific/Auckland'	(GMT+12:00) Auckland, Wellington
87	PACIFIC_TONGATAPU: 'Pacific/Tongatapu'	(GMT+13:00) Nuku'alofa

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var date = new Date(); //Mon Aug 24 2015 17:27:16 GMT-0700 (Pacific Daylight Time)
4   var TOKYO = format.format({
5     value: date,
6     type: format.Type.DATETIME,
7     timezone: format.Timezone.ASIA_TOKYO
8  }); //Returns "8/25/2015 9:27:16 am"
9
10 var NEWYORK = format.format({
11   value: date,
12   type: format.Type.DATETIME,
13   timezone: format.Timezone.AMERICA_NEW_YORK
14 }); //Returns "8/24/2015 8:27:16 pm"
15
16 var dateStr = "03/17/2015 09:00:00 pm"
17   var TOKYO_2 = format.parse({
18     value: dateStr,
19     type: format.Type.DATETIME,
20     timezone: format.Timezone.ASIA_TOKYO
21  }); //Returns Date object [[ Tue Mar 17 2015 05:00:00 GMT-0700 (PDT) ]]
22
23 var NEWYORK_2 = format.parse({
24   value: dateStr,
25   type: format.Type.DATETIME,
26   timezone: format.Timezone.AMERICA_NEW_YORK
27 }); //Returns Date object [[ Tue Mar 17 2015 18:00:00 GMT-0700 (PDT) ]]
28 ...
29 // Add additional code

```

N/format/i18n Module



Note: The content in this help topic pertains to SuiteScript 2.0.

The N/format/i18n module has methods that allows for formatting of strings in international context and formatting of numbers to currency or number strings. It also allows the formatting of phone number to strings and parsing of strings to phone number.

- [N/format/i18n Module Members](#)
- [N/format/i18n Script Samples](#)

N/format/i18n Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	format.CurrencyFormatter	Object	Client and server-side scripts	Represents the object that formats the number to currency string.
	format.NumberFormatter	Object	Client and server-side scripts	Represents the object that formats the number to string.
	format.PhoneNumberFormatter	Object	Client and server-side scripts	Represents the object that formats the phone number to string.
	format.PhoneNumberParser	Object	Client and server-side scripts	Represents the object that parses the string to phone number.
Method	format.spellOut(options)	string	Client and server-side scripts	Creates a string containing the spelled-out version of the specified number in a specified locale.
	format.getCurrencyFormatter(options)	object	Client and server-side scripts	Create the format.CurrencyFormatter object to format numbers to currency strings.
	format.getNumberFormatter(options)	object	Client and server-side scripts	Create the format.NumberFormatter object to format numbers to strings.
Enum	format.NegativeNumberFormat	enum	Client and server-side scripts	Holds the values for the negative number format.
	format.Currency	enum	Client and server-side scripts	Holds the values for the currency code.

Currency Formatter Object Members

The following members are called on the [format.CurrencyFormatter](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	CurrencyFormatter.currency	string	Client and server-side scripts	Indicates the currency code.
	CurrencyFormatter.symbol	string	Client and server-side scripts	Indicates the currency symbol.
	CurrencyFormatter.numberFormatter	object	Client and server-side scripts	Contains the format.NumberFormatter object derived from

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				<code>format.CurrencyFormatter</code> with the same number formatting parameters without currency symbol.
	<code>CurrencyFormatter.format(options)</code>	string	Client and server-side scripts	Formats the number to the currency string.

Number Formatter Object Members

The following members are called on the `format.NumberFormatter` object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	<code>NumberFormatter.groupSeparator</code>	string	Client and server-side scripts	Indicates the group separator.
	<code>NumberFormatter.decimalSeparator</code>	string	Client and server-side scripts	Indicates the decimal separator.
	<code>NumberFormatter.precision</code>	number	Client and server-side scripts	Indicates the precision.
	<code>NumberFormatter.negativeNumberFormat</code>	enum	Client and server-side scripts	Indicates the negative number format.
	<code>NumberFormatter.format(options)</code>	string	Client and server-side scripts	Formats the number to string.

Phone Number Formatter Object Members

The following members are called on the `format.PhoneNumberFormatter` object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	<code>format.PhoneNumberFormatter</code>	Object	Client and server-side scripts	The object that formats the phone number to string.
Method	<code>PhoneNumberFormatter.format(options)</code>	string	Client and server-side scripts	Formats phone number object to string.
Enum	<code>format.PhoneNumberFormatType</code>	Enum	Client and server-side scripts	Holds the values for the phone number format type.
	<code>format.Country</code>	Enum	Client and server-side scripts	Hold the values for the countries.

Phone Number Parser Object Members

The following members are called on the `format.PhoneNumberParser` object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	format.PhoneNumberParser	Object	Client and server-side scripts	The object that parses the string to phone number.
Method	PhoneNumberParser.parse(options)	Object of type PhoneNumber	Client and server-side scripts	Parses string to phone number.

N/format/i18n Script Samples

For help with writing scripts in SuiteScript 2.0, see the help topics [SuiteScript 2.0 Hello World](#) and [SuiteScript 2.0 Entry Point Script Creation and Deployment](#).

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample spells out the number 12345 as a string in German, "zwölftausenddreihundertfünfundvierzig".

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/format/i18n'],
6   function(format) {
7     var spellOut = format.spellOut({
8       number: 12345,
9       locale: "DE"
10    });
11
12   log.debug(spellOut);
13 });

```

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample formats a number to string.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/format/i18n'],
6   function(format) {
7     log.debug("Test of default number formatter:");
8     var numberFormatter = format.getNumberFormatter();
9
10    var gs = numberFormatter.groupSeparator;
11    log.debug("Group separator: " + gs);
12
13    var ds = numberFormatter.decimalSeparator;
14    log.debug("Decimal separator: " + ds);
15
16    var precision = numberFormatter.precision;
17    log.debug("Precision: " + precision);

```

```

18     var nnf = numberFormatter.negativeNumberFormat;
19     log.debug("Negative Number Format: " + nnf);
20
21     log.debug(numberFormatter.format({number: 12.53}));
22     log.debug(numberFormatter.format({number: 12845.22}));
23     log.debug(numberFormatter.format({number: -5421}));
24     log.debug(numberFormatter.format({number: 0.00}));
25     log.debug(numberFormatter.format({number: 0.3456789}));
26
27 });

```

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample formats numbers to currency strings.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/format/i18n'],
6   function(format) {
7     log.debug("Test of currency formatter - EUR:");
8     var curFormatter = format.getCurrencyFormatter({currency: "EUR"});
9
10    var curCur = curFormatter.currency;
11    log.debug("Currency: " + curCur);
12
13    var numberFormat = curFormatter.numberFormatter;
14
15    var cur3 = curFormatter.symbol;
16    log.debug("Currency symbol: " + cur3);
17
18    var c4 = numberFormat.groupSeparator;
19    log.debug("Group separator: " + c4);
20
21    var c5 = numberFormat.decimalSeparator;
22    log.debug("Decimal separator: " + c5);
23
24    var c6 = numberFormat.precision;
25    log.debug("Precision: " + c6);
26
27    var c7 = numberFormat.negativeNumberFormat;
28    log.debug("Negative Number Format: " + c7);
29
30    log.debug(curFormatter.format({number: 12.53}));
31    log.debug(curFormatter.format({number: -5421}));
32    log.debug(curFormatter.format({number: 0.00}));
33    log.debug(curFormatter.format({number: 0.3456789}));
34 });

```

Note: This sample script uses the require function so you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (a script you attach to a script record and deploy). For more information, see the help topic [SuiteScript 2.0 Script Basics](#) [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#) [SuiteScript 2.0 Script Types](#).

```

1 require(['N/format/i18n'],
2   function(format) {
3     log.debug("test parsing from string:");
4     var origNumberStr = "602547854ext.154";
5     log.debug("Original number is " + origNumberStr);
6     var pnParser = format.getPhoneNumberParser({defaultCountry: format.Country.CZECH REPUBLIC});
7     var phoneNumber = pnParser.parse({number: origNumberStr});
8     log.debug("Country code: " + phoneNumber.countryCode);

```

```

9   log.debug("Extension: " + phoneNumber.extension);
10  log.debug("National number: " + phoneNumber.nationalNumber);
11  log.debug("Number of leading zeros: " + phoneNumber.number0fLeadingZeros);
12  log.debug("Carrier code: " + phoneNumber.carrierCode);
13  log.debug("Raw input: " + phoneNumber.rawInput);

14
15  log.debug("\n-----\nFormatting back:");
16  var pnFormatter = format.getPhoneNumberFormatter({});
17  var strNumber = pnFormatter.format({number: phoneNumber});
18  log.debug(strNumber);
19 });

```

Note: This sample script uses the require function so you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (a script you attach to a script record and deploy). For more information, see the help topic [SuiteScript 2.0 Script Basics](#) [SuiteScript 2.0 Script Types](#) [SuiteScript 2.0 Script Types](#).

```

1 require(['N/format/i18n'],
2   function(format) {
3     log.debug("test parsing from string:");
4     var origNumberStr = "7524105210ext.154";
5     log.debug("Original number is " + origNumberStr);
6     var pnParser = format.getPhoneNumberParser({defaultCountry: format.Country.UNITED_STATES});
7     var phoneNumber = pnParser.parse({number: origNumberStr});
8     log.debug(phoneNumber.countryCode);
9     log.debug(phoneNumber.extension);
10    log.debug(phoneNumber.nationalNumber);
11    log.debug(phoneNumber.number0fLeadingZeros);
12    log.debug(phoneNumber.carrierCode);
13    log.debug(phoneNumber.rawInput);

14
15  log.debug("\n-----\ntest formatting back:");
16  var pnFormatter = format.getPhoneNumberFormatter({formatType: format.PhoneNumberFormatType.NATIONAL});
17  var strNumber = pnFormatter.format({number: phoneNumber});
18  log.debug(strNumber);
19 });

```

format.CurrencyFormatter

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The object that formats the number to currency string.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/format/i18n Module
Methods and Properties	N/format/i18n Module Members
Since	2019.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var curFormatter = format.getCurrencyFormatter({currency: "USD"});
6   });
7 // Add additional code

```

CurrencyFormatter.currency

Property Description	Describes the currency code.
Type	string (read-only)
Module	N/format/i18n Module
Parent Object	format.CurrencyFormatter
Sibling Object Members	N/format/i18n Module Members
Since	2019.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var curFormatter = format.getCurrencyFormatter({currency: "USD"});
6     var curCur = curFormatter.currency;
7     log.debug(curCur);
8   });
9 // Add additional code

```

CurrencyFormatter.symbol

Property Description	Describes the symbol of the currency code.
Type	string (read-only)
Module	N/format/i18n Module
Parent Object	format.CurrencyFormatter
Sibling Object Members	N/format/i18n Module Members
Since	2019.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var curFormatter = format.getCurrencyFormatter({currency: "USD"});
6     var curSymbol = curFormatter.symbol;
7     log.debug(curSymbol);
8 });
9 // Add additional code

```

CurrencyFormatter.numberFormatter

Property Description	Contains the <code>format.NumberFormatter</code> object derived from <code>format.CurrencyFormatter</code> with the same number formatting parameters without currency symbol.
Type	string (read-only)
Module	N/format/i18n Module
Parent Object	<code>format.CurrencyFormatter</code>
Sibling Object Members	N/format/i18n Module Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var curFormatter = format.getCurrencyFormatter({currency: "USD"});
6     var numberFormatter = curFormatter.numberFormatter;
7
8     // now numberFormatter object can be used
9     log.debug(numberFormatter.format({number: -12.5366}));
10 });
11 // Add additional code

```

CurrencyFormatter.format(options)

Method Description	Formats the number to the currency string.
Returns	String
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/format/i18n Module
Methods and Properties	N/format/i18n Module Members
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.number	number	required	The number to be formatted

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4         function(format) {
5             var curFormatter = format.getCurrencyFormatter({currency: "USD"});
6             log.debug(curFormatter.format({number: 12.53}));
7 });
8 // Add additional code

```

format.NumberFormatter

Object Description	The object that formats number to string.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/format/i18n Module
Methods and Properties	N/format/i18n Module Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4         function(format) {
5             var numFormatter1 = format.getNumberFormatter(); // no parameter given -> default number formatter object returned
6             var numFormatter2 = format.getNumberFormatter({
7                 groupSeparator: " ",
8                 decimalSeparator: ",",
9                 precision: 2,
10                negativeNumberFormat: format.NegativeNumberFormat_MINUS});
11 // all parameters defined
12
13 // here number formatters can be used
14 log.debug(numFormatter1.format({number: 12.53}));
15 log.debug(numFormatter2.format({number: 12845.22}));

```

```

16 | });
17 | // Add additional code

```

NumberFormatter.groupSeparator

Property Description	Indicates the group separator.
Type	string (read-only)
Module	N/format/i18n Module
Parent Object	format.CurrencyFormatter
Sibling Object Members	N/format/i18n Module Members
Since	2019.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 | // Add additional code
2 | ...
3 | require(['N/format/i18n'],
4 |         function(format) {
5 |             var numFormatter = format.getNumberFormatter({
6 |                 groupSeparator: " ",
7 |                 decimalSeparator: ",",
8 |                 precision: 2,
9 |                 negativeNumberFormat: format.NegativeNumberFormat_MINUS});
10 |            var groupSep = numFormatter.groupSeparator;
11 |        });
12 | // Add additional code

```

NumberFormatter.decimalSeparator

Property Description	Indicates the decimal separator.
Type	string (read-only)
Module	N/format/i18n Module
Parent Object	format.CurrencyFormatter
Sibling Object Members	N/format/i18n Module Members
Since	2019.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 | // Add additional code

```

```

2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var numFormatter = format.getNumberFormatter({
6       groupSeparator: " ",
7       decimalSeparator: ",",
8       precision: 2,
9       negativeNumberFormat: format.NegativeNumberFormat_MINUS});
10    var decimalSep = numFormatter.decimalSeparator;
11  });
12 // Add additional code

```

NumberFormatter.precision

Property Description	Indicates the precision.
Type	number (read-only)
Module	N/format/i18n Module
Parent Object	format.CurrencyFormatter
Sibling Object Members	N/format/i18n Module Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var numFormatter = format.getNumberFormatter({
6       groupSeparator: " ",
7       decimalSeparator: ",",
8       precision: 2,
9       negativeNumberFormat: format.NegativeNumberFormat_MINUS});
10    var precision = numFormatter.precision;
11  });
12 // Add additional code

```

NumberFormatter.negativeNumberFormat

Property Description	Indicates the negative number format.
Type	enum
Module	N/format/i18n Module
Parent Object	format.CurrencyFormatter
Sibling Object Members	N/format/i18n Module Members
Since	2019.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4         function(format) {
5             var numFormatter = format.getNumberFormatter({
6                 groupSeparator: " ",
7                 decimalSeparator: ",",
8                 precision: 2,
9                 negativeNumberFormat: format.NegativeNumberFormat_MINUS});
10            var negNumFormat = numFormatter.negativeNumberFormat;
11        });
12 // Add additional code

```

NumberFormatter.format(options)

Method Description	Format number to the number string.
Returns	String
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/format/i18n Module
Methods and Properties	N/format/i18n Module Members
Since	2019.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.number	number	required	The number to be formatted

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4         function(format) {

```

```

5   var numFormatter = format.getNumberFormatter({
6     groupSeparator: " ",
7     decimalSeparator: ",",
8     precision: 2,
9     negativeNumberFormat: format.NegativeNumberFormat_MINUS));
10 // all parameters defined
11 log.debug(numFormatter.format({number: 12845.22}));
12 });
13 // Add additional code

```

format.PhoneNumberFormatter



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The object that formats the phone number to string.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/format/i18n Module
Methods and Properties	Phone Number Formatter Object Members
Since	2020.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     log.debug("test parsing from string:");
6     var origNumberStr = "602547854ext.154";
7     log.debug("Original number is " + origNumberStr);
8     var pnParser = format.getPhoneNumberParser({
9       defaultCountry: format.Country.CZECH REPUBLIC
10    });
11     var phoneNumber = pnParser.parse({
12       number: origNumberStr
13    });
14     log.debug("Country code: " + phoneNumber.countryCode);
15     log.debug("Extension: " + phoneNumber.extension);
16     log.debug("National number: " + phoneNumber.nationalNumber);
17     log.debug("Number of leading zeros: " + phoneNumber.numberofLeadingZeros);
18     log.debug("Carrier code: " + phoneNumber.carrierCode);
19     log.debug("Raw input: " + phoneNumber.rawInput);

20
21     log.debug("\n-----\nFormatting back:");
22     var pnFormatter = format.getPhoneNumberFormatter({});
23     var strNumber = pnFormatter.format({
24       number: phoneNumber
25     });
26     log.debug(strNumber);
27   });
28 ...
29 // Add additional code

```

format.PhoneNumberParser



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The object that parses the string with the phone number to an object.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/format/i18n Module
Methods and Properties	Phone Number Parser Object Members
Since	2020.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var pnParser = format.getPhoneNumberParser({
6       defaultCountry: format.Country.CZECH REPUBLIC
7     });
8     var origPhoneNumberStr = "602547854ext.154";
9     var phoneNumber = pnParser.parse({
10       number: origPhoneNumberStr
11     });
12     log.debug(origPhoneNumberStr);
13     log.debug(phoneNumber.countryCode);
14     log.debug(phoneNumber.extension);
15     log.debug(phoneNumber.nationalNumber);
16     log.debug(phoneNumber.numberofLeadingZeros);
17     log.debug(phoneNumber.carrierCode);
18     log.debug(phoneNumber.rawInput);
19   });
20 ...
21 // Add additional code

```

PhoneNumberFormatter.format(options)

Method Description	Formats phone number object to string.
Returns	String
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	none
Module	N/format/i18n Module
Parent Object	format.PhoneNumberFormatter
Methods and Properties	Phone Number Formatter Object Members

Since	2020.2
-------	--------

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.number	Object	required	The phone number to be formatted.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     log.debug("test parsing from string:");
6     var origNumberStr = "602547854ext.154";
7     log.debug("Original number is " + origNumberStr);
8     var pnParser = format.getPhoneNumberParser({
9       defaultCountry: format.Country.CZECH REPUBLIC
10    });
11     var phoneNumber = pnParser.parse({
12       number: origNumberStr
13    });
14     log.debug("\nFormatting back:");
15     var pnFormatter = format.getPhoneNumberFormatter({});
16     var strNumber = pnFormatter.format({
17       number: phoneNumber
18    });
19     log.debug(strNumber);
20   });
21 ...
22 // Add additional code

```

PhoneNumberParser.parse(options)

Method Description	Parses the string containing the phone number and returns the phone number object. If the input string contains the country code, it parses the given phone number. If the input string does not contain the country code, the country given to PhoneNumberParser as constructor parameter is used.
Returns	Object
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	none
Module	N/format/i18n Module
Parent Object	format.PhoneNumberParser
Methods and Properties	Phone Number Parser Object Members

Since	2020.2
-------	--------

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.number	string	required	The string to be parsed.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var pnParser = format.getPhoneNumberParser({
6       defaultCountry: format.Country.CZECH REPUBLIC
7     });
8     var phoneNumber = pnParser.parse({
9       number: "602547854ext.154"
10    });
11    log.debug(phoneNumber.countryCode);
12    log.debug(phoneNumber.extension);
13    log.debug(phoneNumber.nationalNumber);
14    log.debug(phoneNumber.numberofLeadingZeros);
15    log.debug(phoneNumber.carrierCode);
16    log.debug(phoneNumber.rawInput);
17  });
18 ...
19 // Add additional code

```

format.PhoneNumberFormatType

Enum Description	Holds the values for the phone number format.
	 Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.
Type	enum
Module	N/format/i18n Module
Sibling Module Members	Phone Number Formatter Object Members
Since	2020.2

Values

- E164

- INTERNATIONAL
- NATIONAL
- RFC3966

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     log.debug("List of valid phone number format types:");
6     for (var fmtType in format.PhoneNumberFormatType) {
7       log.debug(fmtType);
8     }
9   });
10 ...
11 // Add additional code

```

format.Country

Enum Description	Holds the values for the country.
	i Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.
Type	enum
Module	N/format/i18n Module
Sibling Module Members	Phone Number Formatter Object Members
Since	2020.2

Values

All countries.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     log.debug("List of valid countries:");
6     for (var ctr in format.Country) {
7       log.debug(ctr);
8     }

```

```

9   });
10  ...
11 // Add additional code

```

format.spellOut(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Spells out positive and negative number as a string in a specific language For more information, see Codes for the Representation of Names of Languages .
Returns	String
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/format/i18n Module
Methods and Properties	N/format/i18n Module Members
Since	2019.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.number	number	required	The number to be spelled out in a string.	2019.1
options.locale	string	required	<p>The language code that specifies the string's language. ISO 639-1 alpha-2 language codes are supported.</p> <p>The language specified in this parameter is not related to the language specified for a NetSuite account. You can specify any language for this parameter; you do not have to specify a NetSuite supported language.</p> <p>For more information, see Codes for the Representation of Names of Languages.</p>	2019.1

format.getCurrencyFormatter(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Create format.CurrencyFormatter object to format numbers into currency strings.
Returns	Object
Supported Script Types	Client and server-side scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/format/i18n Module
Methods and Properties	N/format/i18n Module Members
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.currency	string	required	Code of the currency that is used by formatter.	2019.2

Errors

Error Code	Thrown If
MISSING_REQD_ARGUMENT	Currency parameter is missing
SSS_INVALID_CURRENCY	The currency is not valid
SSS_INVALID_TYPE_ARG	The parameter type is wrong

format.getNumberFormatter(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Create format.NumberFormatter object to format numbers into strings.
Returns	Object
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/format/i18n Module
Methods and Properties	N/format/i18n Module Members
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.groupSeparator	string	optional	Indicates the group separator.	2019.2

Parameter	Type	Required / Optional	Description	Since
options.decimalSeparator	string	optional	Indicates the decimal separator.	2019.2
options.precision	number	optional	Indicates the precision.	2019.2
options.negativeNumberFormat	enum	optional	Indicates the negative number format.	2019.2

format.NegativeNumberFormat

i Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the negative number format.
Type	enum
Module	N/format/i18n Module
Sibling Module Members	N/format/i18n Module Members
Since	2019.2

Values

Value
BRACKETS
MINUS

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```

1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     var numFormatterM = format.getNumberFormatter({
6       groupSeparator: " ", decimalSeparator: ",",
7       precision: 2,
8       negativeNumberFormat: format.NegativeNumberFormat_MINUS});
9
10    var numFormatterB = format.getNumberFormatter({
11      groupSeparator: " ",
12      decimalSeparator: ",",
13      precision: 2,
14      negativeNumberFormat: format.NegativeNumberFormat_BRACKETS});
15});...

```

```
16 // Add additional code
```

format.Currency

i Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the currency code.
Type	enum
Module	N/format/i18n Module
Sibling Module Members	N/format/i18n Module Members
Since	2019.1

Value

The currency values depend on the company. Examples of currency value include:

- USD
- CAD
- EUR
- GBP
- JPY

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/format/i18n Script Samples](#).

```
1 // Add additional code
2 ...
3 require(['N/format/i18n'],
4   function(format) {
5     log.debug("List of valid currencies:");
6     for (var currency in format.Currency) {
7       log.debug(currency);
8     }
9   });
10 // Add additional code
```

N/http Module

i Note: The content in this help topic pertains to SuiteScript 2.0.

Use the N/http module to make HTTP calls from server or client scripts. For client scripts, this module also provides the ability to make cross-domain HTTP requests using NetSuite servers as proxies.

All HTTP content types are supported.



Note: The N/http module does not accept the HTTPS protocol. Use the [N/https Module](#) for that purpose.

- [HTTP Header Information](#)
- [N/http Module Members](#)
- [ClientResponse Object Members](#)
- [ServerRequest Object Members](#)
- [ServerResponse Object Members](#)
- [N/http Module Script Samples](#)



Important: NetSuite supports the same list of trusted third-party certificate authorities (CAs) as Microsoft. For a list of these CAs, see <http://social.technet.microsoft.com/wiki/contents/articles/31634.microsoft-trusted-root-certificate-program-participants-v-2016-april.aspx>

HTTP Header Information

HTTP headers can be used to pass additional information with an HTTP request or response. Each HTTP header consists of its case-insensitive name followed by a colon (:), then by its value (without line breaks). For a general list of all HTTP headers, visit <http://developer.mozilla.org/en-US/docs/Web/HTTP/Headers>.

Some headers are not supported in NetSuite and are blocked. These are listed below as either general HTTP headers or Suitelet response headers.

General Blocked HTTP Headers

Be aware that certain headers cannot be set manually when using the N/http module methods. If a script attempts to set values for any of the following headers, the values are discarded. These headers are listed in the following table.

<ul style="list-style-type: none"> ■ Connection ■ Content-Length ■ Host ■ JSESSIONID 	<ul style="list-style-type: none"> ■ Trailer ■ Transfer-Encoding ■ Upgrade ■ Via
--	--

Suitelet Response HTTP Header Blocklist

In addition to the headers described in [General Blocked HTTP Headers](#), certain headers cannot be set manually when interacting with the [http.ServerResponse](#) Objects sent by Suitelets. If a script attempts to set values for any of these headers, the system throws an SSS_INVALID_HEADER error. These headers are listed in the following table.

<ul style="list-style-type: none"> ■ Allow ■ Content-Location ■ Content-MD5 ■ Content-Range 	<ul style="list-style-type: none"> ■ Location ■ Proxy-Authenticate ■ Public-Key-Pins ■ Public-Key-Pins-Report-Only 	<ul style="list-style-type: none"> ■ Server ■ Strict-Transport-Security ■ Upgrade-Insecure-Requests ■ Warning
---	--	---

■ Date	■ Retry-After	■ WWW-Authenticate
--------	---------------	--------------------

N/http Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	http.ClientResponse	Object (read-only)	Server scripts	The response from the server to an HTTP client request (e.g., http.get(options)).
	http.ServerRequest	Object (read-only)	Server scripts	HTTP request information sent to an HTTP server. For example, a request sent to/received by a Suitelet or RESTlet.
	http.ServerResponse	Object	Server scripts	The response from an HTTP server to an HTTP request. For example, a response from a Suitelet or RESTlet.
Method	http.delete(options)	http.ClientResponse or http.ServerResponse	Server scripts	Sends an HTTP DELETE request and returns the response.
	http.delete.promise(options)	Promise Object	Client scripts	Sends an HTTP DELETE request asynchronously and returns the response.
	http.get(options)	http.ClientResponse or http.ServerResponse	Server scripts	Sends an HTTP GET request and returns the response.
	http.get.promise(options)	Promise Object	Client scripts	Sends an HTTP GET request asynchronously and returns the response.
	http.post(options)	http.ClientResponse or http.ServerResponse	Server scripts	Sends an HTTP POST request and returns the response.
	http.post.promise(options)	Promise Object	Client scripts	Sends an HTTP POST request asynchronously and returns the response.
	http.put(options)	http.ClientResponse or http.ServerResponse	Server scripts	Sends an HTTP PUT request and returns the response.
	http.put.promise(options)	Promise Object	Client scripts	Sends an HTTP PUT request asynchronously and returns the response.
	http.request(options)	http.ClientResponse or http.ServerResponse	Server scripts	Sends an HTTP request and returns the response.
	http.request.promise(options)	Promise Object	Client scripts	Sends an HTTP request asynchronously and returns the response.
Enum	http.CacheDuration	enum	Server scripts	Holds the string values for supported cache durations. This enum is used to set the value of the ServerResponse.setCdnCacheable(options) property.
	http.Method	enum	Server scripts	Holds the string values for supported HTTP requests. This enum is used to set the value of http.request(options) and ServerRequest.method .
	http.RedirectType	enum	Server scripts	Holds the string values for supported NetSuite resources that you

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				can redirect to. This enum is used to set the value of the type argument for ServerResponse.sendRedirect(options) .

ClientResponse Object Members

The following members are called on the [http.ClientResponse](#) Object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	ClientResponse.body	string (read-only)	Server scripts	The client response body.
	ClientResponse.code	number (read-only)	Server scripts	The client response code.
	ClientResponse.headers	Object (read-only)	Server scripts	The client response headers.

ServerRequest Object Members

The following members are called on the [http.ServerRequest](#) Object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	ServerRequest.getLineCount(options)	number	Server scripts	Returns the number of lines in a sublist.
	ServerRequest.getSublistValue(options)	string	Server scripts	Returns the value of a sublist line item.
Property	ServerRequest.body	string (read-only)	Server scripts	The server request body.
	ServerRequest.files	Object (read-only)	Server scripts	The server request files.
	ServerRequest.headers	Object (read-only)	Server scripts	The server request headers.
	ServerRequest.clientIpAddress	String (read-only)	Server scripts	The remote client IP address.
	ServerRequest.method	http.Method enum	Server scripts	The server request HTTP method.
	ServerRequest.parameters	Object (read-only)	Server scripts	The server request parameters.
	ServerRequest.url	string (read-only)	Server scripts	The server request URL.

ServerResponse Object Members

The following members are called on the [http.ServerResponse](#) Object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	ServerResponse.addHeader(options)	void	Server scripts	Adds a header to the response.
	ServerResponse.getHeader(options)	string string[]	Server scripts	Returns the value of a response header.
	ServerResponse.renderPdf(options)	void	Server scripts	Generates and renders a PDF directly to the response.
	ServerResponse.sendRedirect(options)	void	Server scripts	Sets the redirect URL by resolving to a NetSuite resource.
	ServerResponse.setCdnCacheable(options)	void	Server scripts	Sets CDN caching for a period of time.
	ServerResponse.setHeader(options)	void	Server scripts	Sets the value of a response header.
	ServerResponse.write(options)	void	Server scripts	Writes information (text, xml, html) to the response.
	ServerResponse.writeFile(options)	void	Server scripts	Writes a file to the response.
	ServerResponse.writeLine(options)	void	Server scripts	Writes line information (text, xml, html) to the response.
Property	ServerResponse.headers	Object (read-only)	Server scripts	The server response headers.

N/http Module Script Samples

Example 1

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to use an HTTP GET request for a URL.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/http'], function(http) {
6     function sendGetRequest() {
7         var response = http.get({
8             url: 'http://www.google.com'
9         });
10    }
}

```

```

11     sendGetRequest();
12 });

```

Example 2

i Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

⚠ Important: The value used in this sample for the entity field is a placeholder. Before using this sample, replace the entity field value with a valid value from your NetSuite account. If you run a script with an invalid value, an error may occur.

The following sample shows how to redirect to a new sales order record and set entity to 6. (Assuming there is an entity with number 6, if there's not, then the entity will remain blank.)

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4 */
5 define(['N/record', 'N/http'], function(record, http) {
6     function onRequest(context) {
7         context.response.sendRedirect({
8             type: http.RedirectType.RECORD,
9             identifier: record.Type.SALES_ORDER,
10            parameters: ({
11                entity: 6
12            })
13        });
14    }
15
16    return {
17        onRequest: onRequest
18    };
19});

```

http.ClientResponse

i Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>The response from the server to an HTTP request (e.g., http.get(options)) from a client (i.e., browser).</p> <p>This is the return type for http.delete(options), http.get(options), http.post(options), http.put(options), http.request(options), and corresponding promise methods) when those methods are called from a client (i.e., browser)</p> <p>This object is read-only.</p> <p>For a complete list of this object's properties, see ClientResponse Object Members.</p>
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/http Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var clientResponse = http.get({
4   url: 'http://www.google.com'
5 });
6 ...
7 // Add additional code

```

ClientResponse.body



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The client response body.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ClientResponse
Sibling Object Members	ClientResponse Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var response = http.get({
4   url: 'http://www.google.com'
5 });
6 log.debug({
7   title: 'Client Response Body',
8   details: response.body
9 });
10 ...
11 // Add additional code

```

ClientResponse.code

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The client HTTP response or status code.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ClientResponse
Sibling Object Members	ClientResponse Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var response = http.get({
4   url: 'http://www.google.com'
5 });
6 log.debug({
7   title: 'Client Response Code',
8   details: response.code
9 });
10 ...
11 // Add additional code

```

ClientResponse.headers

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The response header or headers. For more information, see HTTP Header Information .
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/http Module
Parent Object	http.ClientResponse
Sibling Object Members	ClientResponse Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var response = http.get({
4   url: 'http://www.google.com'
5 });
6 log.debug({
7   title: 'Client Response Header',
8   details: response.headers
9 });
10 ...
11 // Add additional code

```

http.ServerRequest

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The HTTP request information set to an HTTP server. For example, a request received by a Suitelet or RESTlet. This object is read-only. For a complete list of this object's methods and properties, see ServerRequest Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 serverRequest.getLineCount({
4   group: 'sublistId'
5 });
6 ...
7 // Add additional code

```

ServerRequest.getLineCount(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the number of lines in a sublist.
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/http Module
Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.group	string	required	The sublist internal ID.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.group parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverRequest.getLineCount({
4   group: 'sublistId'
5 });
6 ...
7 // Add additional code

```

ServerRequest.getSublistValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a sublist line item.
Returns	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/http Module
Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.group	string	required	The sublist internal ID.	2015.2
options.line	string	required	The sublist line number. Note: Sublist index starts at 0.	2015.2
options.name	string	required	The sublist line item ID (name of the field).	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.group or options.line parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverRequest.getSublistValue({
4     group: 'item',
5     name: 'amount',
6     line: '2'
7 });

```

```

8 | ...
9 | //Add additional code

```

ServerRequest.body

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request body.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 | // Add additional code
2 | ...
3 | log.debug({
4 |   title: 'Server Request Body',
5 |   details: request.body
6 | });
7 | ...
8 | // Add additional code

```

ServerRequest.files

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request files.
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module

Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippets show the syntax for this member. They are not functional examples. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug({
4   title: 'Server Request Files',
5   details: request.files
6 });
7 ...
8 // Add additional code

```

```
1 var file = request.files['file_id'];
```

ServerRequest.headers



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The header information included in the http call. This object represents a series of key/value pairs. Each pair represents a request header name and its value.</p> <p>Typically, this object encapsulates two iterations of each header name: one in lower case and another in title case. This behavior is designed so that you can use either lower case or title case when you reference a header. However, the existence of title-case iterations of header names is not guaranteed. For best results, refer to header names using all lower-case letters (and hyphens, when applicable).</p> <p>Important: The request headers and their values are subject to change. If you use these headers in your scripts, you are responsible for testing them to make sure that they contain the information you need. For example, when making an HTTP call to a Suitelet, some headers might be filtered out. Filtering can occur if the headers affect how NetSuite processes the request internally. These filtered headers are not available to the Suitelet, so you should test to see whether a header was filtered out. If so, use a different header instead.</p> <p>For more information, see HTTP Header Information.</p>
Type	Object (read-only)
Supported Script Types	Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug({
4   title: 'Server Request Headers',
5   details: request.headers
6 });
7 ...
8 // Add additional code

```

ServerRequest.clientIpAddress

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The remote client IP address.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 function onRequest(context){
4     var request = context.request;
5     var header = request.headers;
6     var remoteAddress = request.clientIpAddress;
7 }
8 ...
9 // Add additional code

```

ServerRequest.method



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request HTTP method.
Type	http.Method (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug({
4     title: 'Server Request Method',
5     details: request.method
6 });
7 ...
8 // Add additional code

```

ServerRequest.parameters



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request parameters, as name-value pairs. Note that if the server request is a Get request, parameters are sent as part of the URL. If the server request is a Post request, parameters are sent within the request body.
	Note: Parameters cannot be arrays. Use JSON.stringify/JSON.parse instead to handle arrays.
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // example from a Suitelet
4
5 onRequest: function(context) {
6   // in the following, context.request is an http.ServerRequest (per Suitelet onRequest(params.request) entry point)
7
8   if (context.request.method === 'GET') {
9     // request is coming from a User Event script, for instance, on a form load; parameters are in the URL as name values pairs (as the query string)
10
11    var myName = context.request.parameters.custpage_nameParam;           // here, 'custpage_nameParam' and 'custpage_phoneParam' are
12    // parameter names set
13    var myPhone = context.request.parameters.custpage_phoneParam;         // by the requesting script and sent in the HTTP request
14  }
15  if (context.request.method === 'POST'){
16    // request is coming from a Submit button click on the form, submitting the form (via user event script); parameters are in the form fields
17
18    var myName = context.request.parameters.nameFld; // here, 'nameFld' and 'phoneFld' are field ids on the form for the Name and Phone
19    // fields
20    var myPhone = context.request.parameters.phoneFld;
}
}

```

```

21 | ...
22 | // Add additional code

```

ServerRequest.url

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request URL.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 log.debug({
2   title: 'Server Request URL',
3   details: request.url
4 });
5 ...
6 // Add additional code

```

http.ServerResponse

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The response from an HTTP server (e.g., Suitelet or RESTlet) to an HTTP request from a server, such as a user event script. For a complete list of this object's methods and properties, see ServerResponse Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/http Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverResponse.addHeader({
4     name: 'Accept-Language',
5     value: 'en-us',
6 });
7 ...
8 // Add additional code

```

ServerResponse.addHeader(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a header to the response. If the same header has already been set, this method adds another line for that header. For example:
	<pre>1 {Vary: ['Accept-Language', 'Accept-Encoding']}</pre>
	For more information, see HTTP Header Information .
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	The name of the header.	2015.2
options.value	string	required	The value used to set the header.	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_HEADER	One or more headers are not valid.	The header name or value is invalid.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.name or options.value parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverResponse.addHeader({
4   name: 'Accept-Language',
5   value: 'en-us',
6 });
7 ...
8 // Add additional code

```

ServerResponse.getHeader(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value or values of a response header. If multiple values are assigned to the header name, the values are returned as an Array. For more information, see HTTP Header Information .
Returns	string string[]
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	The name of the header.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.name parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverResponse.getHeader({
4   name: 'Accept-Language'
5 });
6 ...
7 // Add additional code

```

ServerResponse.sendRedirect(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the redirect URL by resolving to a NetSuite resource. For example, you could use this method to redirect to a new sales order page for a particular entity.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.identifier	number string	required	The primary ID for this resource. The value you use varies depending on the value of options.type, as follows: ■ MEDIA_ITEM — Use the internal ID of a file stored in the NetSuite File Cabinet.	2015.2

Parameter	Type	Required / Optional	Description	Since
			<ul style="list-style-type: none"> ■ RECORD — Use the record.Type enum to identify the appropriate record type. ■ RESTLET — Use the script ID from the script record of the appropriate RESTlet. ■ SUITELET — Use the script ID from the script record of the appropriate Suitelet. ■ TASK_LINK — Use the appropriate Task ID. Supported IDs are listed in Task IDs. 	
options.type	http.RedirectType	required	The type of resource redirected to.	2015.2
options.editMode	boolean true false	optional	<p>Applicable when redirecting to a record resource.</p> <p>Specifies whether to return a URL for a record in edit mode or view mode.</p> <p>If set to <code>true</code>, returns the record in edit mode. If set to <code>false</code>, returns the record in view mode.</p> <p>The default value is <code>false</code>.</p>	2015.2
options.id	number string	optional	The secondary ID for this resource. If the <code>options.type</code> parameter is set to SUITELET or RESTLET, use the deployment ID. If the <code>options.type</code> parameter is set to RECORD, you can use the internal ID of a specific record instance.	2015.2
options.parameters	Object	optional	Additional URL parameters as name/value pairs.	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_RECORD_TYPE	Type argument <code>{type}</code> is not a valid record or is not available in your account. Please see the documentation for a list of supported record types.	The redirect type is set to record, and an invalid record type is input for <code>options.identifier</code> .
SSS_INVALID_SCRIPT_ID_1	You have provided an invalid script id or internal id: <code>{id}</code>	The type is set to Suitelet or RESTlet, and an invalid script ID or invalid deployment ID is input for <code>options.identifier</code> or <code>options.id</code> .
SSS_INVALID_TASK_ID	The task ID: <code>{id}</code> is not valid. Please refer to the documentation for a list of supported task IDs.	The type is set to task link, and an invalid task ID is input for <code>options.identifier</code> .
SSS_INVALID_URL_CATEGORY	The <code>options.type: {type}</code> is not valid. Please use	The script uses an unrecognizable string value for the <code>options.type</code> parameter. To avoid this error, use http.RedirectType .

Error Code	Message	Thrown If
	http.RedirectType for supported types.	
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.identifier or options.type parameter is not specified. Note that this error is thrown if an enum is misspelled within a script. For example, you see this error if you use http.RedirectType.TASKLINK instead of http.RedirectType.TASK_LINK in the options.type field.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 myServerResponseObj.sendRedirect({
4   type: http.RedirectType.RECORD,
5   identifier: record.Type.SALES_ORDER,
6   parameters: {entity: 8}
7 });
8 ...
9 // Add additional code

```

ServerResponse.setHeader(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of a response header. For more information, see HTTP Header Information .
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	The name of the header.	2015.2

Parameter	Type	Required / Optional	Description	Since
options.value	string	required	The value used to set the header.	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_HEADER	One or more headers are not valid.	The header name or value is invalid.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.name or options.value parameter is not specified.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverResponse.setHeader({
4     name: 'Accept-Language',
5     value: 'en-us',
6 });
7 ...
8 // Add additional code

```

ServerResponse.renderPdf(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Generates and renders a PDF directly to the response.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.xmlString	string	required	Content of the PDF	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.xmlString parameter is not specified.

Syntax

 Important: The following code shows the syntax for this member. It is not a functional example. For a complete script example, see N/http Module Script Samples .
 Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see SuiteScript 2.0 Global Objects .

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4 */
5 define(['N/xml'], function(xml) {
6     return {
7         onRequest: function(context) {
8             var xml = "<?xml version='1.0' encoding='UTF-8'?>\n" +
9                 "<!DOCTYPE pdf PUBLIC '-//big.faceless.org//report\\' \"report-1.1.dtd\\\">\n" +
10                "<pdf lang='ru-RU' xml:lang='ru-RU\\'>\n" +
11                "<head>\n" +
12                "<link name='russianfont' type='font' subtype='opentype' " + "src='NetSuiteFonts/verdana.ttf' " + "src-
bold='NetSuiteFonts/verdanab.ttf' " + "src-italic='NetSuiteFonts/verdanai.ttf' " + "src-bolditalic='NetSuiteFonts/verdan-
abi.ttf' " + "bytes='\\2'\\'/>\n" +
13                "</head>\n" +
14                "<body font-family='russianfont' font-size='18\\'>\nРусский текст</body>\n" +
15                "</pdf>";
16             context.response.renderPdf(xml);
17         }
18     }
19 });

```

ServerResponse.setCdnCacheable(options)

 Note: The content in this help topic pertains to SuiteScript 2.0.
Method Description
Sets CDN caching for a period of time.
Returns
void
Supported Script Types
Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance
None
Module
N/http Module
Parent Object
http.ServerResponse
Sibling Object Members
ServerResponse Object Members

Since	2015.2
--------------	--------

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	enum	required	The value of the caching duration. Set using the http.CacheDuration enum. When used with a Suitelet, if this value is set to UNIQUE, then the Suitelet will never be cached and will always be executed if there's a request to its URL.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.type parameter is not specified.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverResponse.setCdnCacheable({
4   type: http.CacheDuration.MAX
5 });
6 ...
7 // Add additional code

```

ServerResponse.write(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Writes information (text, xml, html) to the response.
	 Note: This method accepts only strings. To pass in a file, you can use ServerResponse.writeFile(options) .
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.output	string	required	The string being written.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.output parameter is not specified.
WRONG_PARAMETER_TYPE	{param name}	The value input for options.output is not a string.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverResponse.write({
4   output: 'Hello World'
5 });
6 ...
7 // Add additional code

```

ServerResponse.writeFile(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Writes a file to the response.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.file	file.File	required	A file.File Object that encapsulates the file to be written.	2015.2
options.isInline	boolean true false	optional	If true , the file is inline. The default value is false.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.file parameter is not specified.
WRONG_PARAMETER_TYPE	{param name}	The value input for options.file is not a file.File Object.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverResponse.writeFile({
4   file: myFileObj,
5   isInline: true
6 });
7 ...
8 // Add additional code

```

ServerResponse.writeLine(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Writes line information (text, xml, html) to the response.
Returns	void

Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.output	string	required	The string being written.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.output parameter is not specified.
WRONG_PARAMETER_TYPE	{param name}	The value input for options.output is not a string.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 serverResponse.writeLine({
4   output: 'this is a sample string'
5 });
6 ...
7 // Add additional code

```

ServerResponse.writePage(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Generates a page.
Returns	void
Supported Script Types	Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.pageObject	serverWidget.Assistant serverWidget.Form serverWidget.List	required	A standalone page Object in the form of an assistant, form, or list.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.pageObject parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPageObj = serverWidget.createList({
4   title: 'Simple List'
5 });
6
7 ServerResponse.writePage({
8   pageObject: myPageObj
9 });
10 ...
11 // Add additional code

```

ServerResponse.headers

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server response headers. For more information, see HTTP Header Information .
-----------------------------	---

Type	Object (read-only) If multiple values are assigned to one header name, the values are returned as an array. For example:
	1 { Vary : ['Accept-Language', 'Accept-Encoding']}
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module
Parent Object	http.ServerResponse
Sibling Object Members	ServerResponse Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug({
4   title: "Server Response Headers",
5   details: ServerResponse.headers
6 });
7 ...
8 // Add additional code

```

http.get(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTP GET request.
Returns	http.ClientResponse or http.ServerResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/http Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.url	string	required	The HTTP URL being requested.	2015.2
options.headers	Object	optional	The HTTP headers. For more information, see HTTP Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An invalid URL is specified in the options.url parameter.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.url parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete HTTP script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = http.get({
8   url: 'http://www.google.com',
9   headers: headerObj
10 });
11 ...
12 // Add additional code

```

http.get.promise(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTP GET request asynchronously.
	Note: The parameters and errors thrown for this method are the same as those for http.get(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	http.get(options)

Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/http Module
Since	2015.2

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4     name: 'Accept-Language',
5     value: 'en-us'
6 };
7 http.get.promise({
8     url: 'http://www.google.com',
9     headers: headerObj
10 })
11     .then(function(response){
12         log.debug({
13             title: 'Response',
14             details: response
15         });
16     })
17     .catch(function onRejected(reason) {
18         log.debug({
19             title: 'Invalid Get Request: ',
20             details: reason
21         });
22     })
23 ...
24 // Add additional code

```

http.delete(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTP DELETE request.  Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	http.ClientResponse or http.ServerResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/http Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.
Note: This method does not include an options.body parameter. Post data is not required when the HTTP method is a DELETE request.

Parameter	Type	Required / Optional	Description	Since
options.url	string	required	The HTTP URL being requested	2015.2
options.headers	Object	optional	The HTTP headers. For more information, see HTTP Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An invalid URL is specified in the options.url parameter.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.url parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/http Module Script Samples .
--

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = http.delete({
8   url: 'http://www.mytestwebsite.com',
9   headers: headerObj
10 });
11 ...
12 // Add additional code

```

http.delete.promise(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Sends an HTTP DELETE request asynchronously.
---------------------------	--

	Note: The parameters and errors thrown for this method are the same as those for http.delete(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	http.delete(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/http Module
Since	2015.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4     name: 'Accept-Language',
5     value: 'en-us'
6 };
7 http.delete.promise({
8     url: 'http://www.mytestwebsite.com',
9     headers: headerObj
10 })
11     .then(function(response){
12         log.debug({
13             title: 'Response',
14             details: response
15         });
16     })
17     .catch(function onRejected(reason) {
18         log.debug({
19             title: 'Invalid Request: ',
20             details: reason
21         });
22     })
23 ...
24 // Add additional code

```

http.post(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTP POST request.
Returns	http.ClientResponse or http.ServerResponse

Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/http Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.body	string Object	required	The POST data.	2015.2
options.url	string	required	The HTTP URL being requested	2015.2
options.headers	Object	optional	The HTTP headers. For more information, see HTTP Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An incorrect protocol is used. For additional information when setting a URL using the N/url module API, see url.resolveScript(options) .
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}.	The options.body or options.url parameter is not specified.
SSS_REQUEST_LOOP_DETECTED	This script executes a recursive function that has exceeded the limit for the number of times a script can call itself using an HTTP request. Please examine the script for a potential infinite recursion problem.	A script is calling back into itself recursively via an HTTP/HTTPS request.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };

```

```

7  var response = http.post({
8    url: 'http://www.testwebsite.com',
9    body: 'My POST Data',
10   headers: headerObj
11 });
12
13 var myresponse_body = response.body; // see http.ClientResponse.body
14 var myresponse_code = response.code; // see http.ClientResponse.code
15 var myresponse_headers = response.headers; // see http.ClientResponse.headers
16 ...
17 // Add additional code

```

http.post.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTP POST request asynchronously.
	 Note: The parameters and errors thrown for this method are the same as those for http.post(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	http.post(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/http Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 http.post.promise({
8   url: 'http://www.google.com',
9   body: 'My POST Data',
10  headers: headerObj
11 })
12 .then(function(response){
13   log.debug({
14     title: 'Response',
15     details: response
16   });
17 })
18 .catch(function onRejected(reason) {

```

```

19 |     log.debug({
20 |       title: 'Invalid Request: ',
21 |       details: reason
22 |     });
23 |
24 ...
25 // Add additional code

```

http.put(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTP PUT request.
	 Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	http.ClientResponse or http.ServerResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/http Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.body	string Object	required	The PUT data.	2015.2
options.url	string	required	The HTTP URL being requested	2015.2
options.headers	Object	optional	The HTTP headers. For more information, see HTTP Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An invalid URL is specified in the options.url parameter.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.body or options.url parameter is not specified.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = http.put({
8   url: 'http://www.google.com',
9   body: 'My PUT Data',
10  headers: headerObj
11 });
12 ...
13 // Add additional code

```

http.put.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Sends an HTTP PUT request asynchronously.</p> <p>Note: The parameters and errors thrown for this method are the same as those for http.put(options). For additional information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	http.put(options)
Supported Script Types	<p>Client scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Client Script Type.</p>
Governance	10 units
Module	N/http Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 http.put.promise({
8   url: 'http://www.google.com',
9   body: 'My PUT Data',
10  headers: headerObj

```

```

11  })
12  .then(function(response){
13    log.debug({
14      title: 'Response',
15      details: response
16    });
17  })
18  .catch(function onRejected(reason) {
19    log.debug({
20      title: 'Invalid Request: ',
21      details: reason
22    });
23  })
24 ...
25 // Add additional code

```

http.request(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTP request.
	 Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	http.ClientResponse or http.ServerResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/http Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.method	enum	required	The HTTP request method. Set using the http.Method enum.  Note: If the method is DELETE, this body data is ignored.	2015.2
options.url	string	required	The HTTP URL being requested	2015.2
options.body	string Object	optional	The POST data if the method is POST. The PUT data if the method is PUT.	
options.headers	Object	optional	The HTTP headers.	2015.2

Parameter	Type	Required / Optional	Description	Since
			For more information, see HTTP Header Information .	

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An invalid URL is specified in the options.url parameter.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.method or options.url parameter is not specified.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = http.request({
8   method: http.Method.GET,
9   url: 'http://www.google.com',
10  body: 'My REQUEST Data',
11  headers: headerObj
12 });
13 ...
14 // Add additional code

```

http.request.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTP request asynchronously.
	 Note: The parameters and errors thrown for this method are the same as those for http.request(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	http.request(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/http Module

Since	2015.2
-------	--------

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4     name: 'Accept-Language',
5     value: 'en-us'
6 };
7 http.request.promise({
8     method: http.Method.GET,
9     url: 'http://www.google.com',
10    body: 'My REQUEST Data',
11    headers: headerObj
12 })
13 .then(function(response){
14     log.debug({
15         title: 'Response',
16         details: response
17     });
18 })
19 .catch(function onRejected(reason) {
20     log.debug({
21         title: 'Invalid Request: ',
22         details: reason
23     });
24 })
25 ...
26 // Add additional code

```

http.CacheDuration

Note: The content in this help topic pertains to SuiteScript 2.0.

Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported cache durations. This enum is used to set the value of the ServerResponse.setCdnCacheable(options) property.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module

Values

■ LONG	Conceptually, this corresponds to days.
■ MEDIUM	Conceptually, this corresponds to hours.
■ SHORT	Conceptually, this corresponds to minutes.

<ul style="list-style-type: none"> ■ UNIQUE 	If UNIQUE is used when working with a Suitelet, then the Suitelet will never be cached and will always be executed if there's a request to its URL.
---	---

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 ServerResponse.setCdnCacheable({
4   type: http.CacheDuration.MEDIUM
5 });
6 ...
7 // Add additional code

```

http.Method

Note: The content in this help topic pertains to SuiteScript 2.0.

Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported HTTP requests. This enum is used to set the value of http.request(options) and ServerRequest.method .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module

Values

- **DELETE**
- **GET**
- **HEAD**
- **PUT**
- **POST**

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var response = http.request({
4   method: http.Method.GET,
5   url: 'http://www.google.com'
6 });
7 ...

```

```
8 // Add additional code
```

http.RedirectType



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported NetSuite resources that you can redirect to. This enum is used to set the value of the type argument for ServerResponse.sendRedirect(options) .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/http Module

Values

- MEDIA_ITEM
- RECORD
- RESTLET
- SUITELET
- TASK_LINK

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/http Module Script Samples](#).

```
1 // Add additional code
2 ...
3 myServerResponseObj.sendRedirect({
4   type: http.RedirectType.RECORD,
5   identifier: record.Type.SALES_ORDER,
6   parameters: {entity: 6}
7 });
8 // Add additional code
```

N/https Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/https module when you need to manage content sent to a third party via HTTPS calls. This module encapsulates all the functionality of the [N/http Module](#), but does not allow the HTTP protocol. You can make HTTPS calls from client and server scripts.

You can use the N/https module to encode binary content or access a handle to the value in a NetSuite credential field.

And, starting with NetSuite 2020.2, you can also use the N/https module to communicate between SuiteScript scripts, RESTlets, and SuiteTalk REST APIs without having to reauthenticate, using the [https.requestRestlet\(options\)](#) and [https.requestSuiteTalkRest\(options\)](#) methods.

When the N/https module is used, SuiteScript also loads the [N/crypto Module](#) and [N/encode Module](#).



Important: Use TLS 1.2 for HTTPS requests. SuiteScript 2.0 requests such [https.delete\(options\)](#), [https.get\(options\)](#), [https.post\(options\)](#), [https.put\(options\)](#), and [https.request\(options\)](#) usually go to third-party servers. Management of these servers is not within the control of your company. These HTTPS requests now fail the handshake when they attempt to connect to servers that do not support TLS 1.2. We recommend that you communicate with those who manage any third-party servers to which you connect, and ensure their servers support the TLS 1.2 protocol.



Important: NetSuite supports the same list of trusted third-party certificate authorities (CAs) as Microsoft. For a list of these CAs, see <http://social.technet.microsoft.com/wiki/contents/articles/31634.microsoft-trusted-root-certificate-program-participants-v-2016-april.aspx>



Warning: Using plain text or other unencrypted user credentials is unsafe and can pose a security threat. Whenever possible, use [Token-based Authentication \(TBA\)](#) or [OAuth 2.0](#) to specify user credentials.

- [HTTPS Header Information](#)
- [N/https Module Members](#)
- [SecureString Object Members](#)
- [ClientResponse Object Members](#)
- [ServerResponse Object Members](#)
- [ServerRequest Object Members](#)
- [N/https Module Script Sample](#)

HTTPS Header Information

HTTPS headers can be used to pass additional information with an HTTPS request or response. Each HTTPS header consists of its case-insensitive name followed by a colon (:), then by its value (without line breaks). For a general list of all HTTP headers (also applicable to HTTPS), visit <http://developer.mozilla.org/en-US/docs/Web/HTTP/Headers>.

Some headers are not supported in NetSuite and are blocked. These are listed below as either general HTTPS headers or Suitelet response headers.

General Blocked HTTPS Headers

Be aware that certain headers cannot be set manually when using N/https module methods. If a script attempts to set values for any of the following headers, the values are discarded. These headers are listed in the following table.

<ul style="list-style-type: none"> ■ Connection ■ Content-Length ■ Host ■ JSESSIONID 	<ul style="list-style-type: none"> ■ Trailer ■ Transfer-Encoding ■ Upgrade ■ Via
--	--

Suitelet Response HTTPS Header Blocklist

In addition to the headers described in [General Blocked HTTPS Headers](#), certain headers cannot be set manually when interacting with the `https.ServerResponse` Objects sent by Suitelets. If a script attempts to set values for any of these headers, the system throws an `SSS_INVALID_HEADER` error. These headers are listed in the following table.

<ul style="list-style-type: none"> ■ Allow ■ Content-Location ■ Content-MD5 ■ Content-Range ■ Date 	<ul style="list-style-type: none"> ■ Location ■ Proxy-Authenticate ■ Public-Key-Pins ■ Public-Key-Pins-Report-Only ■ Retry-After 	<ul style="list-style-type: none"> ■ Server ■ Strict-Transport-Security ■ Upgrade-Insecure-Requests ■ Warning ■ WWW-Authenticate
---	---	---

N/https Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	<code>https.SecureString</code>	Object	Server scripts	Encapsulates data that may be sent to a third-party via an HTTPS call.
	<code>https.ClientResponse</code>	Object (read-only)	Server scripts	Encapsulates the response to an HTTPS client request.
	<code>https.ServerRequest</code>	Object (read-only)	Server scripts	Encapsulates the HTTPS request information sent to an HTTPS server. For example, a request received by a Suitelet or RESTlet.
	<code>https.ServerResponse</code>	Object	Server scripts	Encapsulates the response from an HTTPS server to an HTTPS request. For example, a response from a Suitelet or RESTlet.
Method	<code>https.createSecureKey(options)</code>	Object	Server scripts	Creates a key for the contents of a credential field.
	<code>https.createSecureKey.promise(options)</code>	Promise Object	Client scripts	Creates a key asynchronously for the contents of a credential field.
	<code>https.createSecureString(options)</code>	Object	Server scripts	Creates an <code>https.SecureString</code> Object.
	<code>https.createSecureString.promise(options)</code>	Promise Object	Client scripts	Creates an <code>https.SecureString</code> Object asynchronously.
	<code>https.delete(options)</code>	<code>https.ClientResponse</code> or <code>https.ServerResponse</code>	Server scripts	Sends an HTTPS DELETE request and returns the response.
	<code>https.delete.promise(options)</code>	Promise Object	Client scripts	Sends an HTTPS DELETE request asynchronously and returns the response.
	<code>https.get(options)</code>	<code>https.ClientResponse</code> or <code>https.ServerResponse</code>	Server scripts	Sends an HTTPS GET request and returns the response.
	<code>https.get.promise(options)</code>	Promise Object	Client scripts	Sends an HTTPS GET request asynchronously and returns the response.
	<code>https.post(options)</code>	<code>https.ClientResponse</code> or <code>https.ServerResponse</code>	Server scripts	Sends an HTTPS POST request and returns the response.
	<code>https.post.promise(options)</code>	Promise Object	Client scripts	Sends an HTTPS POST request asynchronously and returns the response.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	https.put(options)	https.ClientResponse or https.ServerResponse	Server scripts	Sends an HTTPS PUT request and returns the response.
	https.put.promise(options)	Promise Object	Client scripts	Sends an HTTPS PUT asynchronously request and returns the response.
	https.request(options)	https.ClientResponse or https.ServerResponse	Server scripts	Sends an HTTPS request and returns the response. If a request fails, an error.SuiteScriptError is thrown.
	https.request.promise(options)	Promise Object	Client scripts	Sends an HTTPS request asynchronously and returns the response. If a request fails, a Promise.reject is thrown with a parameter Error.
	https.requestRestlet(options)	https.ClientResponse	Server scripts	Sends an HTTPS request to a RESTlet and returns the response. Authentication headers are automatically added. The RESTlet will execute with the same privileges as the calling script.
	https.requestSuiteTalkRest(options)	https.ClientResponse	Server scripts	Sends an HTTPS request to a SuiteTalk REST endpoint and returns the response. Authentication headers are automatically added.
Enum	https.CacheDuration	enum	Server scripts	Holds the string values for supported cache durations. This enum is used to set the value of the ServerResponse.setCdnCacheable(options) property.
	https.Encoding	enum	Server scripts	Holds the string values for supported encoding types. This enum is used to set the value of parameters in SecureString.appendString(options) , SecureString.convertEncoding(options) , https.createSecureString(options) .
	https.HashAlg	enum	Server scripts	Holds the string values for supported hashing algorithms. This enum is used to set the value of parameters in SecureString.hash(options) and SecureString.hmac(options) .
	https.Method	enum	Server scripts	Holds the string values for supported HTTP requests. This enum is used to set the value of parameters in https.request(options) and to set the value of ServerRequest.method .
	https.RedirectType	enum	Server scripts	Holds the string values for supported NetSuite resources to which you can redirect. This enum is used to set the value of parameters in ServerResponse.sendRedirect(options) .

SecureString Object Members

SecureString functionality is supported only in server scripts. You can also use this functionality to perform various string transformations using methods that hash, encode, or append another string.

The following members are called on the [https.SecureString](#) Object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	SecureString.appendSecureString(options)	https.SecureString	Server scripts	Appends a passed in https.SecureString to another https.SecureString.
	SecureString.appendString(options)	https.SecureString	Server scripts	Appends a passed in string to a https.SecureString.
	SecureString.convertEncoding(options)	https.SecureString	Server scripts	Changes the encoding of a https.SecureString.
	SecureString.hash(options)	https.SecureString	Server scripts	Produces the https.SecureString as a hash.
	SecureString.hmac(options)	https.SecureString	Server scripts	Produces the https.SecureString as an hmac.

ClientResponse Object Members

The following members are called on the [http.ClientResponse](#) Object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	ClientResponse.body	string (read-only)	Server scripts	The response body.
	ClientResponse.code	number (read-only)	Server scripts	The response code.
	ClientResponse.headers	Object (read-only)	Server scripts	The response body.

ServerRequest Object Members

The following members are called on the [http.ServerRequest](#) Object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	ServerRequest.getLineCount(options)	number	Server scripts	Returns the number of lines in a sublist.
	ServerRequest.getSublistValue(options)	string	Server scripts	Returns the value of a sublist line item.
Property	ServerRequest.body	string (read-only)	Server scripts	The server request body
	ServerRequest.files	Object (read-only)	Server scripts	The server request files.
	ServerRequest.headers	Object (read-only)	Server scripts	The server request headers.
	ServerRequest.method	https.Method enum	Server scripts	The HTTPS method for the server request.
	ServerRequest.parameters	Object (read-only)	Server scripts	The server request parameters.
	ServerRequest.url	string (read-only)	Server scripts	The server request URL.

ServerResponse Object Members

The following members are called on the [http.ServerResponse](#) Object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	ServerResponse.addHeader(options)	void	Server scripts	Adds a header to the response
	ServerResponse.getHeader(options)	string string[]	Server scripts	Returns the value of a response header
	ServerResponse.renderPdf(options)	void	Server scripts	Generates and renders a PDF directly to the response
	ServerResponse.sendRedirect(options)	void	Server scripts	Sets the redirect URL by resolving to a NetSuite resource
	ServerResponse.setCdnCacheable(options)	void	Server scripts	Sets CDN caching for a period of time.
	ServerResponse.setHeader(options)	void	Server scripts	Sets the value of a response header.
	ServerResponse.write(options)	void	Server scripts	Writes information (text/xml/html) to the response.
	ServerResponse.writeFile(options)	void	Server scripts	Writes a file to the response.
	ServerResponse.writeLine(options)	void	Server scripts	Writes line information (text/xml/html) to the response.
Property	ServerResponse.headers	Object (read-only)	Server scripts	The server response headers.

N/https Module Script Sample

i Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to use a GUID to generate a secure token and a secret key. Note this sample is meant to show how to use the APIs, but will not actually work in the debugger because the GUID does not exist in your account. Please try the next sample (which is a Suitelet) for a more complete usage. To run this sample in the debugger, you must replace the GUID with one specific to your account.

```

1 /**
2  * @NApiVersion 2.x
3 */
4

```

```

5  require(['N/https', 'N/crypto', 'N/runtime'], function(http, https, runtime) {
6    function createSecureString() {
7      var passwordGuid = '{284CFB2D225B1D76FB94D150207E49DF}';
8      var secureToken = https.createSecureString({
9        input: passwordGuid
10     });
11     var secretKey = https.createSecretKey({
12       input: passwordGuid
13     });
14     secureToken = secureToken.hmac({
15       algorithm: crypto.HashAlg.SHA256,
16       key: secretKey
17     });
18   }
19   createSecureString();
20 });

```

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to create a form field that generates a GUID using a Suitelet. For more information about credential fields, see [Form.addCredentialField\(options\)](#).

Note: The default maximum length for a credential field is 32 characters. If needed, use the [Field.maxLength](#) property to change this value.

The values for `restrictToDomains`, `restrictToScriptIds`, and `baseUrl` in this sample are placeholders. You must replace them with valid values from your NetSuite account.

```

1  /**
2   * @NApiVersion 2.x
3   * @NScriptType Suitelet
4   */
5
6 define(['N/ui/serverWidget', 'N/https', 'N/url'], function(ui, https, url) {
7   function onRequest(option) {
8     if (option.request.method === 'GET') {
9       var form = ui.createForm({
10         title: 'Password Form'
11       });
12       var credField = form.addCredentialField({
13         id: 'password',
14         label: 'Password',
15         restrictToDomains: ['<accountID>.app.netsuite.com'],
16         restrictToCurrentUser: false,
17         restrictToScriptIds: 'customscript_my_script'
18       });
19       credField.maxLenth = 64;
20       form.addSubmitButton();
21       option.response.writePage({
22         pageObject: form
23       });
24     }
25   else {
26     // Request to an existing suitelet with credentials
27     var passwordGuid = option.request.parameters.password;
28
29     // Replace SCRIPTID and DEPLOYMENTID with the internal ID of the suitelet script and deployment in your account
30     var baseUrl = url.resolveScript({
31       scriptId: SCRIPTID,
32       deploymentId: DEPLOYMENTID,
33       returnExternalURL: true
34     });
35     var authUrl = baseUrl + '&pwd={' + passwordGuid + '}';
36     var secureStringUrl = https.createSecureString({

```

```

37     input: authUrl
38   });
39   var secureStringPWD = https.createSecureString({
40     input: '{' + passwordGuid + '}'
41   });
42   var headers = ({
43     'pwd': secureStringPWD
44   });
45   var response = https.post({
46     credentials: [passwordGuid],
47     url: secureStringUrl,
48     body: {authorization: ' ' + secureStringPWD + '', data:'anything can be here'},
49     headers: headers
50   });
51 }
52 }
53
54 return {
55   onRequest: onRequest
56 };
57 });

```

https.SecureString



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates data that may be sent to a third-party via an HTTPS call, such as a fragment of sensitive data. This object is needed when you create a SecureString, put your data in it, and encode it a particular way. For a complete list of this object's methods, see SecureString Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 function createSecureString() {
4   var passwordGuid = '(284CFB2D225B1D76FB94D150207E49DF)';
5   var secureToken = https.createSecureString({
6     input: passwordGuid
7   });
8 }
9 ...
10 // Add additional code

```

For additional information on specifying user credentials, you may also want to visit the help topics: [Token-based Authentication \(TBA\)](#) and [OAuth 2.0](#).

SecureString.appendSecureString(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Appends the passed in https.SecureString to another https.SecureString .
Returns	https.SecureString
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.secureString	https.SecureString	required	The https.SecureString to append.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 string1.appendSecureString({
4   secureString: secureString2
5 });
6 ...
7 // Add additional code

```

SecureString.appendString(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Appends the passed string to an https.SecureString .
Returns	https.SecureString
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.input	string	required	The string to append.
options.inputEncoding	https.Encoding	required	The encoding of the string that is being appended.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 string1.appendString({
4   input: '48656c6c6f20776f726c640d0a',
5   encoding: https.Encoding.HEX);
6 ...
7 // Add additional code

```

SecureString.convertEncoding(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Changes the encoding of a https.SecureString .
Returns	https.SecureString
Governance	None
Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.toEncoding	https.Encoding	required	The encoding to apply to the returned string.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code ...
2 https.convertEncoding({
3   toEncoding: https.Encoding.HEX
4 });
5 ...
6 // Add additional code

```

SecureString.hash(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Hashes an https.SecureString Object
Returns	https.SecureString
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/https Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.algorithm	https.HashAlg	required	The hash algorithm. Set the value using the https.HashAlg enum.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 secureString = secureString.hash({
4   algorithm: crypto.HashAlg.SHA256
5 });
6 ...
7 // Add additional code

```

SecureString.hmac(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Produces the securestring as an hmac.
Returns	https.SecureString
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.algorithm	https.HashAlg	required	The hash algorithm. Set by the https.HashAlg enum.
options.key	crypto.SecretKey	required	A key returned from https.createSecureKey(options) .

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 secureToken = secureToken.hmac({
4   algorithm: crypto.HashAlg.SHA256,
5   key: secretKey
6 });
7 ...
8 // Add additional code

```

https.createSecureKey(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates and returns a crypto.SecretKey Object. This method can take a GUID. Use Form.addCredentialField(options) to generate a value.
---------------------------	---

	You can put the key in your secure string. SuiteScript decrypts the value (key) and sends it to the server
Returns	crypto.SecretKey
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.guid	string	required	A GUID used to generate a secret key. The GUID can resolve to either data or metadata.	2015.2
options.encoding	https.Encoding	optional	Specifies the encoding for the SecureKey.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var secretKey = https.createSecureKey({
4   encoding: https.Encoding.HEX,
5   guid: '284CFB2D225B1D76FB94D150207E49DF'
6 });
7 ...
8 // Add additional code

```

https.createSecureKey.promise(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates and returns a crypto.SecretKey Object asynchronously.
	Note: The parameters and errors thrown for this method are the same as those for https.createSecureKey(options) . For more information on promises, see Promise Object .

Synchronous Version	https.createSecureKey(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/https Module
Since	2015.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var secretKey = https.createSecretKey.promise({
4   encoding: https.Encoding.HEX,
5   guid: '284CFB2D225B1D76FB94D150207E49DF'
6 });
7 ...
8 // Add additional code
9

```

https.createSecureString(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates and returns an https.SecureString .
Returns	https.SecureString
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.input	string	required	The string to convert to a https.SecureString .	Release 15 Version 2

Parameter	Type	Required / Optional	Description	Since
options.inputEncoding	https.Encoding	optional	Identifies the encoding that the input string uses. The default value is UTF_8.	Release 15 Version 2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var secureToken = https.createSecureString({
4   input: passwordGuid
5 });
6 ...
7 // Add additional code

```

https.createSecureString.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates and returns an https.SecureString asynchronously.
	Note: The parameters and errors thrown for this method are the same as those for https.createSecureString(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	https.createSecureString(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/https Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var secureToken = https.createSecureString.promise({
4   input: passwordGuid
5 });

```

```

6 | ...
7 | // Add additional code

```

https.ClientResponse

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the response to an HTTPS client request. This object is read-only. For a complete list of this object's properties, see ClientResponse Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var clientResponse = https.get({
4   url: 'https://www.testwebsite.com'
5 });
6 ...
7 // Add additional code

```

ClientResponse.body

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The client response body.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var response = https.get({
4   url: 'https://www.testwebsite.com'
5 });
6 log.debug({
7   title: 'Client Response Body',
8   details: response.body
9 });
10 ...
11 // Add additional code

```

ClientResponse.code



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The client HTTP response or status code.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var response = https.get({
4   url: 'https://www.testwebsite.com'
5 });
6 log.debug({
7   title: 'Client Response Code',
8   details: response.code
9 });
10 ...
11 // Add additional code

```

ClientResponse.headers

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The response header or headers. For more information, see HTTPS Header Information .
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var response = https.get({
4   url: 'https://www.testwebsite.com'
5 });
6 log.debug({
7   title: 'Client Response Header',
8   details: response.headers
9 });
10 ...
11 // Add additional code

```

https.ServerRequest

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The HTTPS request information set to an HTTPS server. For example, a request received by a Suitelet or RESTlet. This object is read-only. For a complete list of this object's methods and properties, see ServerRequest Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module

Since	2015.2
--------------	--------

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverRequest.getLineCount({
4     group: 'sublistId'
5 });
6 ...
7 // Add additional code

```

ServerRequest.getLineCount(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the number of lines in a sublist.
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.group	string	required	The sublist internal ID.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.group parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```
1 // Add additional code
```

```

2 ...
3 serverRequest.getLineCount({
4   group: 'sublistId'
5 });
6 ...
7 // Add additional code

```

ServerRequest.getSublistValue(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a sublist line item.
Returns	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.group	string	required	The sublist internal ID.	2015.2
options.line	string	required	The sublist line number. Sublist index starts at 0.	2015.2
options.name	string	required	The name of the field.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.group, options.line, or the options.name parameter is not specified.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverRequest.getSublistValue({
4   group: 'item',
5   name: 'amount',

```

```

6   line: '2'
7 });
8 ...
9 // Add additional code

```

ServerRequest.body



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request body.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 log.debug({
4   title: 'Server Request Body',
5   details: request.body
6 });
7 ...
8 // Add additional code

```

ServerRequest.files



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request files.
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module

Since	2015.2
--------------	--------

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. They are not functional examples. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 log.debug({
4   title: 'Server Request Files',
5   details: request.files
6 });
7 ...
8 // Add additional code

```

```
1 var file = request.files['file_id'];
```

ServerRequest.headers



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>This object represents a series of key/value pairs. Each pair represents a server request header name and its value.</p> <p>Typically, this object encapsulates two iterations of each header name: one in lower case and another in title case. This behavior is designed so that you can use either lower case or title case when you reference a header. However, the existence of title-case iterations of header names is not guaranteed. For best results, refer to header names using all lower-case letters (and hyphens, when applicable).</p> <p>Important: The server request headers and their values are subject to change. If you use these headers in your scripts, you are responsible for testing them to make sure that they contain the information you need. For example, when making an HTTP call to a Suitelet, some headers might be filtered out. Filtering can occur if the headers affect how NetSuite processes the request internally. These filtered headers are not available to the Suitelet, so you should test to see whether a header was filtered out. If so, use a different header instead.</p> <p>For more information, see HTTPS Header Information.</p>
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 log.debug({
4   title: 'Server Request Headers',
5   details: request.headers
6 });
7 ...
8 // Add additional code

```

ServerRequest.method

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request HTTPS method.
Type	enum (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 log.debug({
4   title: 'Server Request Method',
5   details: request.method
6 });
7 ...
8 // Add additional code

```

ServerRequest.parameters

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request parameters, as name-value pairs. Note that if the server request is a Get request, parameters are sent as part of the URL. If the server request is a Post request, parameters are sent within the request body.
	Note: Parameters cannot be arrays. Use JSON.stringify/JSON.parse instead to handle arrays.
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Parent Object	https.ServerRequest
Sibling Object Members	ServerRequest Object Members
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // example from a Suitelet
4
5 onRequest: function(context) {
6     // in the following, context.request is an https.ServerRequest (per Suitelet onRequest(params.request) entry point)
7
8     if (context.request.method === 'GET') {
9         // request is coming from a User Event script; for instance, on a form load; parameters are in the URL as name values pairs (as the query string)
10
11         var myName = context.request.parameters.custpage_nameParam;           // here, 'custpage_nameParam' and 'custpage_phoneParam' are
12         var myPhone = context.request.parameters.custpage_phoneParam;          // by the requesting script and sent in the HTTPS request
13     }
14     if (context.request.method === 'POST'){
15         //request is coming from a Submit button click on the form, submitting the form (via user event script); parameters are in the form fields
16
17         var myName = context.request.parameters.nameFld;    // here, 'nameFld' and 'phoneFld' are field ids on the form for the Name and Phone
18         var myPhone = context.request.parameters.phoneFld;
19     }
20 }
21 ...

```

```
22 // Add additional code
```

ServerRequest.url



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The server request URL.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```
1 // Add additional code
2 ...
3 log.debug({
4   title: 'Server Request URL',
5   details: request.url
6 });
7 ...
8 // Add additional code
```

https.ServerResponse



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The response from an HTTPS server (e.g., Suitelet or RESTlet) to an HTTPS request from a server, such as a user event script. For a complete list of this object's methods and properties, see ServerResponse Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverResponse.addHeader({
4   name: 'Accept-Language',
5   value: 'en-us',
6 });
7 ...
8 // Add additional code

```

ServerResponse.addHeader(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a header to the response. If the same header has already been set, this method adds another line for that header. For example:
	<pre>1 {Vary: ['Accept-Language', 'Accept-Encoding']}</pre> For more information, see HTTPS Header Information .
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	The name of the header.	2015.2
options.value	string	required	The value used to set the header.	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_HEADER	One or more headers are not valid.	The header name or value is invalid.

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.name or options.value parameter is not specified.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverResponse.addHeader({
4     name: 'Accept-Language',
5     value: 'en-us',
6 });
7 ...
8 // Add additional code

```

ServerResponse.getHeader(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value or values of a response header. If multiple values are assigned to the header name, the values are returned as an Array. For more information, see HTTPS Header Information .
Returns	string string[]
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	The name of the header.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.name parameter is not specified.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverResponse.getHeader({
4     name: 'Accept-Language'
5 });
6 ...
7 // Add additional code

```

ServerResponse.renderPdf(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Generates and renders a PDF directly to the response.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.xmlString	string	required	Content of the pdf.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.xmlString parameter is not specified.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...

```

```

3 | serverResponse.renderPDF({
4 |   xmlString:'<?xml version="1.0"?>\n<!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">\n<pdf>\n<body font-
5 |   size="18">\nHello World!\n</body>\n</pdf>'
6 | });
7 | // Add additional code

```

ServerResponse.sendRedirect(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a redirect URL that resolves to a NetSuite resource. For example, you could use this method to redirect to a new sales order page for a particular entity.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

 **Important:** All parameters must be prefixed with custparam.

Parameter	Type	Required / Optional	Description	Since
options.identifier	number string	required	The primary ID for this resource. The value you use varies depending on the value of options.type, as follows: <ul style="list-style-type: none">■ MEDIA_ITEM — Use the internal ID of a file stored in the NetSuite File Cabinet.■ RECORD — Use the record.Type enum to identify the appropriate record type.■ RESTLET — Use the script ID from the script record of the appropriate RESTlet.■ SUITELET — Use the script ID from the script record of the appropriate Suitelet.■ TASK_LINK — Use the appropriate Task ID. Supported IDs are listed in Task IDs.	2015.2
options.type	string	required	The type of resource to which the script redirects. Use the https.RedirectType enum to set a value for this parameter.	2015.2
options.editMode	boolean true false	optional	Applicable when redirecting to a record resource.	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p>Specifies whether to return a URL for a record in edit mode or view mode.</p> <p>If set to true , returns the record in edit mode. If set to false , returns the record in view mode.</p> <p>The default value is false.</p>	
options.id	string	optional	The secondary ID for this resource. If the options.type parameter is set to SUITELET or RESTLET, use the deployment ID. If the options.type parameter is set to RECORD, you can use the internal ID of a specific record instance.	2015.2
options.parameters	object	optional	Additional URL parameters as key-value pairs.	2015.2

Errors

Error Code	Message	Thrown If
INVALID_ID	You have provided an invalid script id or internal id: {id}	The options.type parameter is set to RESTLET or SUITELET, and the script uses an invalid ID for options.identifier or options.id.
INVALID_RCRD_TYPE	The record type {type} is invalid.	The options.type parameter is set to RECORD, and the script uses an unrecognizable string value for options.identifier. To avoid this error, use the record.Type enum to identify the appropriate record type.
INVALID_TASK_ID	The task ID: {id} is not valid. Please refer to the documentation for a list of supported task IDs.	The options.type parameter is set to TASK_LINK, and the script uses an invalid task ID for options.identifier. For a list of valid IDs, see the help topic Task IDs .
SSS_INVALID_URL_CATEGORY	The options.type: {type} is not valid. Please use the https.RedirectType enum for supported types.	The script uses an unrecognizable string value for the options.type parameter. To avoid this error, use the https.RedirectType enum.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.identifier or options.type parameter is not specified. Note that this error is thrown if an enum is misspelled within a script. For example, you see this error if you use http.RedirectType.TASKLINK instead of http.RedirectType.TASK_LINK in the options.type field.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 myServerResponseObj.sendRedirect({
4   type: https.RedirectType.RECORD,
```

```

5   identifier: record.Type.SALES_ORDER,
6   parameters: {entity: 8}
7 });
8 ...
9 //Add additional code

```

ServerResponse.setCdnCacheable(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets CDN caching for a period of time.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	https.CacheDuration .	required	The value of the caching duration. When used with a Suitelet, if this value is set to UNIQUE, then the Suitelet will never be cached and will always be executed if there's a request to its URL.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.type parameter is not specified.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverResponse.setCdnCacheable({
4   type: https.CacheDuration.LONG

```

```

5 });
6 ...
7 // Add additional code

```

ServerResponse.setHeader(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of a response header. For more information, see HTTPS Header Information .
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	The name of the header.	2015.2
options.value	string	required	The value used to set the header.	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_HEADER	One or more headers are not valid.	The header name or value is invalid.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.name or options.value parameter is not specified.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverResponse.setHeader({
4   name: 'Accept-Language',
5   value: 'en-us',
6 });

```

```

7 | ...
8 | //Add additional code

```

ServerResponse.write(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Writes information (text, xml, html) to the response. Note: This method accepts only strings. To pass in a file, you can use ServerResponse.writeFile(options) .
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.output	string	required	The string being written.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.output parameter is not specified.
WRONG_PARAMETER_TYPE	{param name}	The value input for options.output is not a string.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverResponse.write({
4   output: 'Hello World'
5 });
6 ...

```

```
7 // Add additional code
```

ServerResponse.writeFile(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Writes a file to the response.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.file	file.File	required	A file.File Object that encapsulates the file to be written.	2015.2
options.isInline	boolean true false	optional	Determines whether the field is inline. If true, the file is inline.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.file parameter is not specified.
WRONG_PARAMETER_TYPE	{param name}	The value input for options.file is not a file.File Object.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```
1 // Add additional code
2 ...
3 serverResponse.writeFile({
4   file: myFileObj,
5   isInline: true
6 });
7 ...
8 // Add additional code
```

ServerResponse.writeLine(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Writes line information (text, xml, html) to the response.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.output	string	required	The string being written.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.output parameter is not specified.
WRONG_PARAMETER_TYPE	{param name}	The value input for options.output is not a string.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 serverResponse.writeLine({
4     output: 'this is a sample string'
5 });
6 ...
7 // Add additional code

```

ServerResponse.writePage(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Generates a page.
Returns	void

Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/https Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.pageObject	serverWidget.Assistant serverWidget.Form serverWidget.List	required	A standalone page Object in the form of an assistant, form or list.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.pageObject parameter is not specified.

Syntax

 Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/https Module Script Sample .
--

```

1 // Add additional code
2 ...
3 var myPageObj = serverWidget.createList({
4   title: 'Simple List'
5 });
6
7 ServerResponse.writePage({
8   pageObject: myPageObj
9 });
10 ...
11 // Add additional code

```

ServerResponse.headers

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Property Description	The server response headers. For more information, see HTTPS Header Information .
Type	Object (read-only) Note that If multiple values are assigned to one header name, the values are returned as an array. For example:

	<pre>1 {Vary: ['Accept-Language', 'Accept-Encoding']}</pre>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module
Since	2015.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You attempted to edit this property. This property is read-only.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```
1 // Add additional code
2 ...
3 log.debug({
4   title: "Server Response Headers",
5   details: serverResponse.headers
6 });
7 ...
8 // Add additional code
```

https.get(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS GET request.
Returns	https.ClientResponse or https.ServerResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.url	string	required	The HTTPS URL being requested	2015.2

Parameter	Type	Required / Optional	Description	Since
options.headers	Object	optional	The HTTPS headers. For more information, see HTTPS Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An invalid URL is specified in the options.url parameter.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.url parameter is not specified.
SSS_REQUEST_LOOP_DETECTED	This script executes a recursive function that has exceeded the limit for the number of times a script can call itself using an HTTP request. Please examine the script for a potential infinite recursion problem.	A script is calling back into itself recursively via an HTTP/HTTPS request.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = https.get({
8   url: 'https://www.testwebsite.com',
9   headers: headerObj
10 });
11 ...
12 // Add additional code

```

https.get.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS GET request asynchronously.
	 Note: The parameters and errors thrown for this method are the same as those for https.get(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	https.get(options)
Supported Script Types	Client scripts

	For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/https Module
Since	2015.2

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4     name: 'Accept-Language',
5     value: 'en-us'
6 };
7 https.get.promise({
8     url: 'https://www.testwebsite.com',
9     headers: headerObj
10 })
11     .then(function(response){
12         log.debug({
13             title: 'Response',
14             details: response
15         });
16     })
17     .catch(function onRejected(reason) {
18         log.debug({
19             title: 'Invalid Get Request: ',
20             details: reason
21         });
22     })
23 ...
24 // Add additional code

```

https.delete(options)

ℹ Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS DELETE request.
⚠ Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.	
Returns	https.ClientResponse or https.ServerResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.url	string	required	The HTTPS URL being requested	2015.2
options.headers	Object	optional	The HTTPS headers. For more information, see HTTPS Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An invalid URL is specified in the options.url parameter.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.url parameter is not specified.
SSS_REQUEST_LOOP_DETECTED	This script executes a recursive function that has exceeded the limit for the number of times a script can call itself using an HTTP request. Please examine the script for a potential infinite recursion problem.	A script is calling back into itself recursively via an HTTP/HTTPS request.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/https Module Script Sample .
--

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = https.delete({
8   url: 'https://www.mytestwebsite.com',
9   headers: headerObj
10 });
11 ...
12 // Add additional code

```

https.delete.promise(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Sends an HTTP DELETE request asynchronously.
---------------------------	--

	Note: The parameters and errors thrown for this method are the same as those for https.delete(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	https.delete(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/https Module
Since	2015.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4     name: 'Accept-Language',
5     value: 'en-us'
6 };
7 https.delete.promise({
8     url: 'https://www.mytestwebsite.com',
9     headers: headerObj
10 })
11     .then(function(response){
12         log.debug({
13             title: 'Response',
14             details: response
15         });
16     })
17     .catch(function onRejected(reason) {
18         log.debug({
19             title: 'Invalid Request: ',
20             details: reason
21         });
22     })
23 ...
24 // Add additional code

```

https.post(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS POST request.
Returns	https.ClientResponse or https.ServerResponse

Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.body	string Object	required	The POST data.	2015.2
options.url	string	required	The HTTPS URL being requested.	2015.2
options.credentials	string[]	optional	An array of string GUIDs. These GUIDs are searched for in the options.body of the request and are replaced by actual decrypted passwords before they are sent to a third-party server.	2015.2
options.headers	Object	optional	The HTTPS headers. For more information, see HTTPS Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An incorrect protocol is used, such as using HTTP within the HTTPS module. For additional information when setting a URL using the N/url module API, see url.resolveScript(options) .
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}.	The options.body or options.url parameter is not specified.
SSS_REQUEST_LOOP_DETECTED	This script executes a recursive function that has exceeded the limit for the number of times a script can call itself using an HTTP request. Please examine the script for a potential infinite recursion problem.	A script is calling back into itself recursively via an HTTP/HTTPS request.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```
1 // Add additional code
```

```

2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = https.post({
8   url: 'https://www.testwebsite.com',
9   body: 'My POST Data',
10  headers: headerObj
11 });
12
13 var myresponse_body = response.body; //see https.ClientResponse.body
14 var myresponse_code = response.code; //see https.ClientResponse.code
15 var myresponse_headers = response.headers; //see https.ClientResponse.headers
16 ...
17 //Add additional code

```

https.post.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS POST request asynchronously.
	Note: The parameters and errors thrown for this method are the same as those for https.post(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	https.post(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/https Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };https.post.promise({
7   url: 'https://www.testwebsite.com',
8   body: 'My POST Data',
9   headers: headerObj
10 })
11 .then(function(response){
12   log.debug({

```

```

13     title: 'Response',
14     details: response
15   });
16 }
17 .catch(function onRejected(reason) {
18   log.debug({
19     title: 'Invalid Request: ',
20     details: reason
21   });
22 });
23 ...
24 // Add additional code

```

https.put(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS PUT request.
	Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	https.ClientResponse or https.ServerResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.body	string Object	required	The PUT data.	2015.2
options.url	string	required	The HTTPS URL being requested.	2015.2
options.credentials	string[]	optional	An array of string GUIDs. These GUIDs are searched for in the options.body of the request and are replaced by actual decrypted passwords before they are sent to a third-party server.	2015.2
options.headers	Object	optional	The HTTPS headers. For more information, see HTTPS Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An invalid URL is specified in the options.url parameter.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.body or options.url parameter is not specified.
SSS_REQUEST_LOOP_DETECTED	This script executes a recursive function that has exceeded the limit for the number of times a script can call itself using an HTTP request. Please examine the script for a potential infinite recursion problem.	A script is calling back into itself recursively via an HTTP/HTTPS request.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = https.put({
8   url: 'https://www.testwebsite.com',
9   body: 'My PUT Data',
10  headers: headerObj
11 });
12 ...
13 // Add additional code

```

https.put.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS PUT request asynchronously. <div style="background-color: #e0f2ff; padding: 10px;"> i Note: The parameters and errors thrown for this method are the same as those for https.put(options). For more information on promises, see Promise Object. </div>
Returns	Promise Object
Synchronous Version	https.put(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units

Module	N/https Module
Since	2015.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4     name: 'Accept-Language',
5     value: 'en-us'
6 };
7 https.put.promise({
8     url: 'https://www.testwebsite.com',
9     body: 'My PUT Data',
10    headers: headerObj
11 })
12 .then(function(response){
13     log.debug({
14         title: 'Response',
15         details: response
16     });
17 })
18 .catch(function onRejected(reason) {
19     log.debug({
20         title: 'Invalid Request: ',
21         details: reason
22     });
23 })
24 ...
25 // Add additional code

```

https.request(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS request.
	<p>Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.</p>
Returns	https.ClientResponse or https.ServerResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.method	enum	required	The HTTPS request method. Set using the https.Method enum.	2015.2
options.url	string	required	The HTTPS URL being requested	2015.2
options.body	string Object	optional	The POST data if the method is POST. The PUT data if the method is PUT.  Note: If the method is DELETE, this body data is ignored.	2015.2
options.credentials	string[]	optional	An array of string GUIDs. These GUIDs are searched for in the options.body of the request and are replaced by actual decrypted passwords before they are sent to a third-party server.	2015.2
options.headers	Object	optional	The HTTPS headers. For more information, see HTTPS Header Information .	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL	The URL must be a fully qualified HTTP/HTTPS URL.	An invalid URL is specified in the options.url parameter.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.method or options.url parameter is not specified.
SSS_REQUEST_LOOP_DETECTED	This script executes a recursive function that has exceeded the limit for the number of times a script can call itself using an HTTP request. Please examine the script for a potential infinite recursion problem.	A script is calling back into itself recursively via an HTTP/HTTPS request.

Syntax

 Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/https Module Script Sample .
--

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 var response = https.request({
8   method: https.Method.GET,

```

```

9 |     url: 'https://www.testwebsite.com',
10|     body: 'My REQUEST Data',
11|     headers: headerObj
12| });
13| ...
14| // Add additional code

```

https.request.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS request asynchronously.
	 Note: The parameters and errors thrown for this method are the same as those for https.request(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	https.request(options)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/https Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var headerObj = {
4   name: 'Accept-Language',
5   value: 'en-us'
6 };
7 https.request.promise({
8   method: https.Method.GET,
9   url: 'https://www.testwebsite.com',
10  body: 'My REQUEST Data',
11  headers: headerObj
12
13 })
14 .then(function(response){
15   log.debug({
16     title: 'Response',
17     details: response
18   });
19 })
20 .catch(function onRejected(reason) {
21   log.debug({
22     title: 'Invalid Request: ',
23     details: reason
24   });
25 })

```

```

26 | ...
27 | //Add additional code

```

https.requestRestlet(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS request to a RESTlet and returns the response. Authentication headers are automatically added. The RESTlet will run with the same privileges as the calling script.
Returns	https.ClientResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https Module
Since	2020.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.body	string Object	required, if options.method = POST or PUT.	The PUT/POST data. This is ignored if the options.method is not POST or PUT.	2020.2
options.deploymentId	string	required	The script ID of the script deployment record.	2020.2
options.scriptId	string number	required	The internal ID or script ID of the script record. Specify internal ID as a number. Specify script ID as a string.	2020.2
options.headers	Object	optional	The HTTPS headers.	2020.2
options.method	string	optional	The HTTPS method (DELETE, GET, HEAD, POST, PUT). The default value is GET if options.body is not specified, and POST if options.body is specified.	2020.2
options.urlParams	Object	optional	The parameters to be appended to the target URL as a query string.	2020.2

Errors

Error Code	Message	Thrown If
INVALID_SCRIPT_DEPLOYMENT_ID_1		If the options.deploymentId parameter does not reference a valid deployment for the script.

Error Code	Message	Thrown If
SSS_AUTHORIZATION_HEADER_NOT_ALLOWED		The authorization header is set.
SSS_INVALID_HEADER		The options.headers parameter is in an invalid format or contains an invalid header.
SSS_INVALID_SCRIPT_ID_1		The options.scriptId parameter does not reference a RESTlet script.
SSS_INVALID_URL_PARAMS		The options.urlParams parameter is in an invalid format.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.body, options.deploymentID, or options.scriptID parameter is not specified.
SSS_REQUEST_LOOP_DETECTED		The script executes a recursive function that has exceeded the limit for the number of times a script can call itself using an HTTPS request.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var myRestletHeaders = {
4   myHeaderType: 'Test',
5   myHeaderStuff: 'This is my header',
6   myHeaderId: 7
7 };
8 var myUrlParameters = {
9   myFirstParameter: 'firstparam',
10  mySecondParameter: 'secondparam'
11 }
12 var myRestletResponse = https.requestRestlet({
13   body: 'My Restlet body',
14   deploymentId: 'deploy1',
15   headers: myRestletHeaders,
16   method: 'GET',
17   scriptId: 99,
18   urlparams: myUrlParameters
19 });
20 Add additional code

```

https.requestSuiteTalkRest(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sends an HTTPS request to a SuiteTalk REST endpoint and returns the response. Authentication headers are automatically added.
Returns	https.ClientResponse
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	10 units
Module	N/https Module
Since	2020.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.body	string Object	required, if options.method = POST or PUT	The PUT/POST data. This is ignored if the options.method parameter is not POST or PUT.	2020.2
options.url	string	required	The URL of a SuiteTalk REST endpoint. It may also contain query parameters. The URL may be fully qualified, relative, or relative with the /services/rest/ prefix omitted.	2020.2
options.headers	Object	optional	The HTTPS headers.	2020.2
options.method	string	optional	The HTTPS method (DELETE, GET, HEAD, POST, PUT). The default value is GET if options.body is not specified, and POST if options.body is specified.	2020.2

Errors

Error Code	Message	Thrown If
SSS_AUTHORIZATION_HEADER_NOT_ALLOWED		The authorization header is set.
SSS_INVALID_HEADER		The options.headers parameter is in an invalid format or contains an invalid header.
SSS_INVALID_URL		If the value of the options.url parameter is invalid or does not reference a SuiteTalk REST endpoint.
SSS_MISSING_REQD_ARGUMENT	Missing a required argument: {param name}	The options.body, options.method, or options.url parameter is not specified.
SSS_REQUEST_LOOP_DETECTED		The script executes a recursive function that has exceeded the limit for the number of times a script can call itself using an HTTPS request.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```
1 | // Add additional code
```

```

2 ...
3 var mySuiteTalkRestHeaders = {
4   myHeaderType: 'Test',
5   myHeaderStuff: 'This is my header',
6   myHeaderId: 7
7 };
8 var myRestSuiteTalkRestResponse = https.requestSuiteTalkRest({
9   body: 'My SuiteTalk Rest body',
10  headers: mySuiteTalkRestHeaders,
11  method: GET,
12  url: 'www.SuiteTalkRestUrl.com'
13 });
14 // Add additional code

```

https.CacheDuration

Note: The content in this help topic pertains to SuiteScript 2.0.

Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported cache durations. This enum is used to set the value of the ServerResponse.setCdnCacheable(options) property.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module

Values

■ LONG	Conceptually, this corresponds to days.
■ MEDIUM	Conceptually, this corresponds to hours.
■ SHORT	Conceptually, this corresponds to minutes.
■ UNIQUE	If UNIQUE is used when working with a Suitelet, then the Suitelet will never be cached and will always be executed if there's a request to its URL.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 ServerResponse.setCdnCacheable({
4   type: https.CacheDuration.LONG
5 });
6 ...
7 // Add additional code

```

https.Encoding



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported encoding values.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module

Values

- UTF_8
- BASE_16
- BASE_32
- BASE_64
- BASE_64_URL_SAFE
- HEX

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var mySecretKey = https.createSecretKey({
4   encoding: https.Encoding.HEX,
5   guid: '284CFB2D225B1D76FB94D150207E49DF'
6 });
7 ...
8 // Add additional code

```

https.HashAlg



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported hashing algorithms.
-------------------------	---

Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module

Values

- SHA1
- SHA256
- SHA512
- MD5

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var mySecureString = https.createSecureString({
4   input: 'ConvertMe'
5 });
6 var mySecureStringHash = mySecureString.hash({
7   algorithm: https.HashAlg.SHA256
8 });
9 ...
10 // Add additional code

```

https.Method

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

 **Note:** JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported HTTPS requests. This enum is used to set the value of https.request(options) and ServerRequest.method .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module

Values

- DELETE
- GET
- HEAD
- PUT

■ POST

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var response = https.request({
4   method: https.Method.GET,
5   url: 'https://www.testwebsite.com'
6 });
7 ...
8 // Add additional code

```

https.RedirectType



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported NetSuite resources that you can redirect to. This enum is used to set the value of the type argument for ServerResponse.sendRedirect(options) .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/https Module

Values

Value	Description
MEDIA_ITEM	A file in the NetSuite File Cabinet
RECORD	A NetSuite record.
RESTLET	A deployed RESTlet.
SUITELET	A deployed Suitelet.
TASK_LINK	A page in NetSuite, as defined by a valid Task ID.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/https Module Script Sample](#).

```
1 // Add additional code
```

```

2 ...
3 myServerResponseObj.sendRedirect({
4   type: https.RedirectType.RECORD,
5   identifier: record.Type.SALES_ORDER,
6   parameters: {entity: 6}
7 });
8 // Add additional code

```

N/https/clientCertificate Module

i Note: The content in this help topic pertains to SuiteScript 2.0.

Load the clientCertificate module to send SSL requests with a digital certificate.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	clientCertificate. post(options)	https.ClientResponse	Server-side scripts	Sends a SSL secured POST request to a remote server.
	clientCertificate. get(options)	https.ClientResponse	Server-side scripts	Sends a SSL secured GET request to a remote server.
	clientCertificate. put(options)	https.ClientResponse	Server-side scripts	Sends a SSL secured PUT request to a remote server.
	clientCertificate. delete(options)	https.ClientResponse	Server-side scripts	Sends a SSL secured DELETE request to a remote server.
	clientCertificate. request(options)	https.ClientResponse	Server-side scripts	Sends a SSL secured REQUEST request to a remote server.

N/https/clientCertificate Module Script Sample

The following is an example of how to send a certificate to a Brazilian tax authority for authentication.

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample sends a secure post request to a remote URL.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/https/clientCertificate'],function (cert){
6   var url = "https://nfe.fazenda.sp.gov.br/ws/cadconsultacadastro4.asmx";

```

```

7   var data = "<?xml version='1.0' encoding='utf-8'?><soapenv:Envelope xmlns:soapenv='http://www.w3.org/2003/05/soap-
envelope'><soapenv:Body><ns1:nfeDadosMsg xmlns:ns1='http://www.portalfiscal.inf.br/nfe/wsdl/CadConsultaCadastro4'><Con-
sCad xmlns='http://www.portalfiscal.inf.br/nfe' versao='2.00'><infCons><xServ>CONS-CAD</xServ><UF>SP</UF><CNPJ>47508411000156</
CNPJ></infCons> </ConsCad></ns1:nfeDadosMsg></soapenv:Body></soapenv:Envelope>";
8   var key = "custcertificate1";
9   var headers = {
10     "Content-Type": "application/soap+xml"
11   };
12
13   var response = cert.post({
14     url: url,
15     certId: key,
16     body: data,
17     headers: headers
18   });
19   log.debug(response.body);
20 })

```

clientCertificate.post(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to send a SSL secured POST request to a remote service and return the response.
	Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	An https.ClientResponse Object
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https/clientCertificate Module
Since	2019.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required/Optional	Description	Since
options.url	string	required	The URL address of the remote server.	2019.1
options.certId	string	required	The ID of the client certificate.	2019.1
options.body	string	required	The POST data to be sent to the remote server.	2019.1
options.headers	object	optional	The HTTPS headers associated with the request.	2019.1

clientCertificate.get(options)

<p>Note: The content in this help topic pertains to SuiteScript 2.0.</p>	
Method Description	Method used to send a SSL secured GET request to a remote service and return the response. Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	An https.ClientResponse Object
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https/clientCertificate Module
Since	2019.2

Parameters

<p>Note: The options parameter is a JavaScript object.</p>				
Parameter	Type	Required/Optional	Description	Since
options.url	string	required	The URL address of the remote server.	2019.2
options.certId	string	required	The ID of the client certificate.	2019.2
options.headers	object	optional	The HTTPS headers associated with the request.	2019.2

clientCertificate.put(options)

<p>Note: The content in this help topic pertains to SuiteScript 2.0.</p>	
Method Description	Method used to send a SSL secured request to a remote service and return the response. Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	An https.ClientResponse Object
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units

Module	N/https/clientCertificate Module
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required/Optional	Description	Since
options.url	string	required	The URL address of the remote server.	2019.2
options.body	string	required	The PUT data to be sent to the remote server.	2019.2
options.certId	string	required	The ID of the client certificate.	2019.2
options.headers	object	optional	The HTTPS headers associated with the request.	2019.2

clientCertificate.delete(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to send a SSL secured request to a remote service and return the response.
	 Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	An https.ClientResponse Object
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https/clientCertificate Module
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required/Optional	Description	Since
options.url	string	required	The URL address of the remote server.	2019.2
options.certId	string	required	The ID of the client certificate.	2019.2

options.headers	object	optional	The HTTPS headers associated with the request.	2019.2
-----------------	--------	----------	--	--------

clientCertificate.request(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to send a SSL secured request to a remote service and return the response.  Important: If negotiating a connection to the destination server exceeds 5 seconds, a connection timeout occurs. If transferring a payload to the server exceeds 45 seconds, a request timeout occurs.
Returns	An https.ClientResponse Object
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/https/clientCertificate Module
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required/Optional	Description	Since
options.url	string	required	The URL address of the remote server.	2019.2
options.body	string	required for PUT and POST methods optional for HEAD, GET, DELETE	The REQUEST data to be sent to the remote server.	2019.2
options.certId	string	required	The ID of the client certificate.	2019.2
options.headers	object	required	The HTTP headers associated with the request.	2019.2
options.method	string	required	The HTTP method to be used. Use the https.Method enum to set this value.	2019.2

N/keyControl Module

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

The N/keyControl module can access key storage, which is also available in the UI at Setup > Company > Preferences > Keys. By using the SSH keys, you can manage files and directories by using the SSH file transfer (SFTP) protocol. For more information, see the help topic [SSH Keys for SFTP](#).

For more information about SFTP, see [N/sftp Module](#).

N/keyControl Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	keyControl.Key	Object	Server-side scripts	Represents the key object.
Method	keyControl.findKeys(options)	Object	Server-side scripts	Searches and returns a list of keys based on criteria set. If no options are set for criteria, the full list of keys stored in NetSuite is returned.
	keyControl.createKey(options)	keyControl.Key	Server-side scripts	Creates a key.
	keyControl.deleteKey(options)	Object	Server-side scripts	Marks the key as deleted in database. The history is retained.
	keyControl.loadKey(options)	Object	Server-side scripts	Loads a key.
Enum	keyControl.Operator	enum	Server-side scripts	Holds the values for key operators.

Key Object Members

The following members are called on the [keyControl.Key](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	Key.file	file.File	Server-side scripts	File object of the key.
	Key.password	Object	Server-side scripts	Password of the key (write-only). You can create a GUID using Form.addSecretKeyField(options) .
	Key.scriptId	Object	Server-side scripts	Script ID of the key.
	Key.name	Object	Server-side scripts	Name of the key.
	Key.description	Object	Server-side scripts	Description of the key.
	Key.restrictions	Object	Server-side scripts	The internal IDs of the employees selected in the Restrict to Employees field of the key record.
Method	Key.save()	Object	Server-side scripts	Saves the key.

N/keyControl Module Script Sample

Example 1

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how you can create a key.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/keyControl','N/file'],function(keyControl,file){
6      var key = keyControl.createKey();
7      key.file = file.load(422);
8      //id of file containing private key (id_ecdsa or id_rsa)
9      key.name = "SFTP key";
10     key.save();
11 })

```

Example 2

Note: This script sample uses the `define` function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the `require` function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how you can add a secret key field.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4  */
5 define(['N/ui/serverWidget', 'N/file', 'N/keyControl', 'N/runtime'], function(ui, file, keyControl, runtime) {
6     function onRequest(context) {
7         var request = context.request;
8         var response = context.response;
9
10        if (request.method === 'GET') {
11            var form = ui.createForm({
12                title: 'Enter Password'
13            });
14
15            var credField = form.addSecretKeyField({
16                id: 'custfield_password',
17                label: 'Password',
18                restrictToScriptIds: [runtime.getCurrentScript().id],
19                restrictToCurrentUser: true //Depends on use case
20            });
21            credField.maxLength = 64;
22
23            form.addSubmitButton();
24            response.writePage(form);
25        } else {
26            // Read the request parameter matching the field ID we specified in the form
27            var passwordToken = request.parameters.custfield_password;

```

```

28     var pem = file.load({
29         id: 422
30     });
31
32     var key = keyControl.createKey();
33     key.file = pem;
34     key.name = 'Test';
35     key.password = passwordToken;
36     key.save();
37 }
38 }
39 return {
40     onRequest: onRequest
41 };
42 });
43 });

```

keyControl.Key



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Represents the key.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/keyControl Module
Methods and Properties	Key Object Members
Since	2019.2

Syntax

```

1 // Add additional code
2 ...
3 var key = keyControl.createKey();
4     key.file = file.load(422);
5     //id of file containing private key (id_ecdsa or id_rsa)
6 ...
7 // Add additional code

```

Key.file



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The file object of the key.
Type	file.File
Module	N/keyControl Module
Parent Object	keyControl.Key
Supported Script Types	All server-side scripts

	For additional information, see the help topic SuiteScript 2.0 Script Types .
Methods and Properties	Key Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var key = keyControl.createKey();
4 key.file = file.load(422);
5 ...
6 // Add additional code

```

Key.password



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The password of the key. GUID or secret token for working with passwords is accepted.
Type	String (write-only)
Module	N/keyControl Module
Parent Object	keyControl.Key
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Methods and Properties	Key Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var passwordToken = request.parameters.custfield_password;
4     var pem = file.load(id:422);
5     var key = keyControl.createKey();
6     key.file = pem;
7     key.name = 'Test';
8     key.password = passwordToken;
9 ...
10 // Add additional code

```

Key.scriptId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The script ID of the key. Using Key.save() and keyControl.findKeys(options) returns the script ID.
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/keyControl Module
Methods and Properties	Key Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var key = keyControl.createKey();
4 key.scriptId = 'testid'
5 ...
6 // Add additional code

```

Key.name



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of the key.
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/keyControl Module
Methods and Properties	Key Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var key = keyControl.createKey();
4 key.name = 'testname'

```

```

5 | ...
6 | // Add additional code

```

Key.description



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The description of the key.
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/keyControl Module
Methods and Properties	Key Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 | // Add additional code
2 | ...
3 | var key = keyControl.createKey();
4 | key.description = 'testdescription'
5 | ...
6 | // Add additional code

```

Key.restrictions



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of employee IDs. Only these employees can access the key.
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/keyControl Module
Methods and Properties	Key Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 | // Add additional code

```

```

2 ...
3 var key = keyControl.createKey();
4 key.restriction = 'testrestrictions'
5 ...
6 //Add additional code

```

Key.save()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Saves the key.
Returns	Object
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	Key Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var key = keyControl.createKey();
4 key.file = file.load(422);
5 //id of file containing private key
6 key.name = 'SFTP key';
7 key.save();
8 ...
9 //Add additional code

```

keyControl.createKey(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a key record on the Private Keys page using a file from the File Cabinet.
Returns	keyControl.Key
Supported Script Types	Server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/keyControl Module
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.file	file	optional	The file with the key.	2019.2
options.password	string	optional	The password that is associated with the key.	2019.2
options.scriptId	string	optional	The script ID for the newly-created key.	2019.2
options.description	string	optional	The description of the key.	2019.2
options.restrictions	number[] or string[]	optional	The array of employee internal IDs selected in the Restricted to Employees field for a key.	2019.2
options.name	string	optional	The name of the key.	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 require(['N/keyControl','N/file'],function(keyControl,file){
4     var key = keyControl.createKey();
5     key.file = file.load(422);
6     //id of file containing private key (id_ecdsa or id_rsa)
7     key.name = "SFTP key";
8     key.save();
9 })
10 ...
11 // Add additional code

```

keyControl.findKeys(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a list of keys that are available to the user.
Returns	Metadata about the keys
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/keyControl Module
Since	2019.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.restriction	number	optional	The internal ID of an employee selected in the Restrict to Employees field.	2019.2
options.name	string or object	optional	<p>The name of the key. The properties of the object are:</p> <ul style="list-style-type: none"> ■ value is a string, which can be used if object is used instead of string. ■ operator is one of the operator enum. ■ ignoreCase is either true or false. <p>If the object is used, the value is mandatory. Operator defaults to equals and ignoreCase defaults to true.</p>	2019.2
options.description	string or object	optional	<p>The description of the key. The properties of the object are:</p> <ul style="list-style-type: none"> ■ value is a string, which can be used if object is used instead of string. ■ operator is one of the operator enum. ■ ignoreCase is either true or false. <p>If the object is used, the value is mandatory. Operator defaults to equals and ignoreCase defaults to true.</p>	2019.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 require(['N/keyControl'],function(keyControl){
4     var keys = keyControl.findKeys({
5         name:{value: 'test',
6             operator: keyControl.Operator.CONTAINS, ignoreCase:true}
7     });
8 ...
9 // Add additional code

```

keyControl.deleteKey(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description

Deletes a key.

Returns	Object
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/keyControl Module
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.scriptId	string	required	The script ID of the key to be deleted. Using Key.save() and keyControl.findKeys(options) returns the script ID.	2019.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var keyId = keyControl.deleteKey({
4  scriptId: 'key_test'
5 });
6 ...
7 // Add additional code

```

keyControl.loadKey(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Loads a key.
Returns	Object
Supported Script Types	Server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/keyControl Module
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.scriptId	string	required	The script ID of the key to be loaded. Using Key.save() and keyControl.findKeys(options) returns the script ID.	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var key = keyControl.loadKey({
4  scriptId: '_testKey'
5 });
6 ...
7 // Add additional code

```

keyControl.Operator



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the key operators of keyControl.findKeys(options) .
Module	N/keyControl Module
Supported Script Types	All server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Since	2019.2

Values

Value	Sets Property To
STARTS_WITH	startswith
CONTAINS	contains

ENDS_WITH	endswith
EQUALS	equals

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/keyControl Module Script Sample](#).

```

1 require(['N/keyControl'],function(keyControl){
2   var keys = keyControl.findKeys({
3     name:{
4       value: 'test',
5       operator: keyControl.Operator.CONTAINS
6   });
7 })

```

N/log Module



Note: The content in this help topic pertains to SuiteScript 2.0.

The log methods for logging script execution details can be accessed globally or by loading the N/log module. Load the N/log module when you want to manually access its members, such as for testing purposes. For more information about the global log object, see [log Object](#).

Log messages appear on the Execution Log tab of the script deployment for deployed scripts, or on the Execution Log tab of the SuiteScript Debugger if you are debugging a script. Log messages also appear on the Script Execution Logs page at Customization > Scripting > Script Execution Logs.

- [N/log Module Members](#)
- [N/log Module Guidelines](#)
- [Using Log Levels](#)
- [Viewing Script Execution Logs](#)
- [N/log Module Script Sample](#)
- [Governance on Script Logging](#)

N/log Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	log.audit(options)	void	Client and server scripts	Logs an Audit message.
	log.debug(options)	void	Client and server scripts	Logs a Debug message.
	log.emergency(options)	void	Client and server scripts	Logs an Emergency message.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	log.error(options)	void	Client and server scripts	Logs an Error message.

N/log Module Guidelines

The following guidelines are provided for use of the N/log module:

- NetSuite governs the amount of logging that can be done in any specific 60 minute time period. A company is allowed to make up to 100,000 log object method calls across **all** of their scripts. Script owners are notified if NetSuite detects that one script is logging excessively and automatically adjusts the log level.
- NetSuite purges logs older than **30 days**.
- The Server Script Log search and the Execution Log tab of script records and script deployment records have a log storage limit of 5 million per database. Because log persistence is not guaranteed, NetSuite recommends using custom records if you want to store script execution logs for extended periods. The Script Execution Logs page at Customization > Scripting > Script Execution Logs does not share the same database limit.
- The Execution Log tab also lists notes returned by NetSuite such as error messages. For more information, see [N/error Module](#).
- If you deploy a client script to a form using [Form.clientScriptFileId](#) or [Form.clientScriptModulePath](#), using the N/log module adds the logs to the deployment of the parent script. The parent script can be either a beforeLoad user event script or a [SuiteScript 2.0 Suitelet Script Type](#).
- When an object (that is not a string) is passed to a log object method, NetSuite runs [JSON.stringify\(obj\)](#) on any values that are passed as the details parameter and equal a JavaScript object.

```

1 | ...
2 | //log.debug(myRecord) //Shows the JSON representation of the current values in the myRecord object
3 | var id = myRecord.save();
4 | ...

```

Using Log Levels

Use the log methods along with the Log Level field on the script deployment to specify which log messages are written to the Execution Log on the script deployment. This is useful during the debugging of a script or for providing useful execution notes for auditing or tracking purposes.

Log levels on the script deployment and N/log module methods act as a filter on the amount of information logged. The following Log Levels are supported:

Log Level	Example Uses
Debug	<p>Use Debug to show all messages.</p> <p>This type of logging is suitable only for testing scripts. To avoid excessive logging, the Debug log level is not recommended for active scripts in production.</p>
Audit	Use Audit to shows a record of events that have occurred during the processing of the script (for example, "A request was made to an external site.").
Error	Use Error to show only unexpected script errors.
Emergency	Use Emergency to show only the most critical errors in the script log.

The following table shows you which type of log messages are written based on the Log Level field selected on the script deployment and the log method used in your script.

Log Level	Log method used in your script			
	log.emergency(options)	log.error(options)	log.audit(options)	log.debug(options)
Emergency	X			
Error	X	X		
Audit	X	X	X	
Debug	X	X	X	X

Viewing Script Execution Logs

Log messages for a specific script are shown on the Execution Log of the script deployment for the script. These logs are not guaranteed to persist for 30 days and may be purged to enhance performance if volume is high.

To view script execution log details for various scripts, go to Customization > Scripting > Script Execution Logs. This list of script execution logs is an enhanced repository that stores all log details for 30 days.

On this page, you can perform the following tasks:

- Search for specific logs using filter options, such as log level, execution date range, and script name.
- Download the list as a CSV file or an Excel spreadsheet.
- Print the list.

When you debug a script in the SuiteScript Debugger, log messages appear on the Execution Log tab of the SuiteScript Debugger and do not appear on any Execution Tab of any script deployment.

N/log Module Script Sample

i Note: This sample script uses the define function, which is used in your entry point script (the script you attach to a script record). Keep in mind that you must use the require function instead of define if you want to copy your script into the debugger and test it.

The following sample shows how to create each type of log messages.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType UserEventScript
4 */
5
6 define(['N/log'],function(log){
7     function beforeLoad(context){
8         var myValue = 'value';
9
10        var myObject = {
11            name: 'Jane',
12            id: '123'
13        };
14
15        log.audit({
16            title: 'Audit Entry',
17            details: myObject
18        });

```

```

19 |     log.debug({
20 |       title: 'Debug Entry',
21 |       details: 'Value of myValue is: ' + myValue
22 |     });
23 |     log.emergency({
24 |       title: 'Emergency Entry'
25 |     });
26 |     log.error({
27 |       title: 'Error Entry',
28 |       details: 'Value of myValue is: ' + myValue
29 |     });
30 |   }
31 |   return {
32 |     beforeLoad: beforeLoad
33 |   };
34 | });

```

log.audit(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Logs an Audit message. Audit messages appear on the Execution Log tab if the Log Level on the script deployment is set to Audit or Debug. Use this method for scripts in production.
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Amount of logging in any 60 minute period is limited. See N/log Module Guidelines .
Module	N/log Module
Since	2016.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	required	The text to appear in the Title column on the Execution Log tab of the script deployment. Maximum length is 99 characters. If you set this value to null, an empty string (''), or omit it, the word "Untitled" appears for the log entry.	2016.1
options.details	any	optional	You can pass any value for this parameter. If the value is a JavaScript Object type, JSON.stringify(obj) is called on the object before displaying the value. NetSuite truncates any resulting string over 3999 characters.	2016.1

Syntax

The following code snippet shows the syntax for this method.

```

1 //Add additional code
2 ...
3 var myValue = 'value';
4 log.audit({
5   title: 'Audit Entry',
6   details: 'Value of myValue is: ' + myValue
7 });
8 ...
9 //Add additional code

```

log.debug(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Logs a Debug message. Debug messages appear on the Execution tab only if the Log Level on the script deployment is set to Debug. Use this method for scripts in development.
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Amount of logging in any 60 minute period is limited. See N/log Module Guidelines .
Module	N/log Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	required	The text to appear in the Title column on the Execution Log tab of the script deployment. Maximum length is 99 characters. If you set this value to null, an empty string (""), or omit it, the word "Untitled" appears for the log entry.	2016.1
options.details	any	optional	The text of the log message, that can include strings, variables, objects, etc. If the value is a JavaScript object type, JSON.stringify(obj) is called on the object before displaying the value.	2016.1

Parameter	Type	Required / Optional	Description	Since
			NetSuite truncates any resulting string over 3999 characters.	

Syntax

The following code snippet shows the syntax for this method.

```

1 //Add additional code
2 ...
3 var myValue = 'value';
4 log.debug({
5   title: 'Debug Entry',
6   details: 'Value of myValue is: ' + myValue
7 });
8 ...
9 //Add additional code

```

log.emergency(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Logs an Emergency message. Emergency messages appear on the Execution tab only if the Log Level on the script deployment is set to Audit, Debug, Error, or Emergency. In other words, emergency messages always appear. Use this method for scripts in production.
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Amount of logging in any 60 minute period is limited. See N/log Module Guidelines .
Module	N/log Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	required	The text to appear in the Title column on the Execution Log tab of the script deployment. Maximum length is 99 characters. If you set this value to null, an empty string (""), or omit it, the word "Untitled" appears for the log entry.	2016.1
options.details	any	optional	The text of the log message, that can include strings, variables, objects, etc.	2016.1

Parameter	Type	Required / Optional	Description	Since
			If the value is a JavaScript Object type, JSON.stringify(obj) is called on the object before displaying the value. NetSuite truncates any resulting string over 3999 characters.	

Syntax

The following code snippet shows the syntax for this method.

```

1 //Add additional code
2 ...
3 var myValue = 'value';
4 log.emergency({
5   title: 'Emergency Entry',
6   details: 'Value of myValue is: ' + myValue
7 });
8 ...
9 //Add additional code

```

log.error(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Logs an Error message. Error messages appear on the Execution tab only if the Log Level on the script deployment is set to Audit, Debug, or Error. Use this method for scripts in production.
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Amount of logging in any 60 minute period is limited. See N/log Module Guidelines .
Module	N/log Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	required	The text to appear in the Title column on the Execution Log tab of the script deployment. Maximum length is 99 characters. If you set this value to null, an empty string (''), or omit it, the word "Untitled" appears for the log entry.	2016.1

Parameter	Type	Required / Optional	Description	Since
options.details	any	optional	<p>The text of the log message, that can include strings, variables, objects, etc.</p> <p>If the value is a JavaScript object type, JSON.stringify(obj) is called on the object before displaying the value.</p> <p>NetSuite truncates any resulting string over 3999 characters.</p>	2016.1

Syntax

The following code snippet shows the syntax for this method.

```

1 //Add additional code
2 ...
3 var myValue = 'value';
4 log.error({
5   title: 'Error Entry',
6   details: 'Value of myValue is: ' + myValue
7 });
8 ...
9 //Add additional code

```

N/piremoval Module

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Load the N/piremoval module to remove personal information (PI) from system notes, workflow history, and specific field values. Use the N/piremoval module to comply with the General Data Protection Regulation (GDPR), specifically the right to be forgotten. You can remove personal information from system notes only, or you can also remove workflow history and field values on the record. Entity records, transactions, and custom records are supported.

You can use the [piremoval.createTask\(options\)](#) method to create a PI removal task, or use [piremoval.loadTask\(options\)](#) to load an existing PI removal task. Both of these methods return a [piremoval.PiRemovalTask](#) object that represents the task. Create a [piremoval.PiRemovalTask](#) object for each record type that requires removal of personal information. Use the [PiRemovalTask.save\(\)](#) method to save the task, then use the [PiRemovalTask.run\(\)](#) method to process the task and remove the personal information.

You can use the [piremoval.getTaskStatus\(options\)](#) method to check the status of a submitted PI removal task. This method returns a [piremoval.PiRemovalTaskStatus](#) object that describes the current status of the removal task. The [piremoval.PiRemovalTaskStatus](#) object uses an iterator to provide a list of log entries in the [PiRemovalTaskStatus.logList](#) object.

To use the N/piremoval module, the following requirements must be met:

- Remove Personal Information Create permission is required to create a PI removal task.
- Remove Personal Information Run permission is required to run a PI removal task.

For more information, see the help topic [Personal Information \(PI\) Removal](#).

In this help topic

- [N/piremoval Module Members](#)

- PiRemovalTask Object Members
- PiRemovalTaskLogItem Object Members
- PiRemovalTaskStatus Object Members
- N/piremoval Module Script Samples

N/piremoval Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	piremoval.PiRemovalTask	Object	Server scripts	Encapsulates a personal information removal task. Use piremoval.createTask(options) to create this object.
	piremoval.PiRemovalTaskLogItem	Object	Server scripts	Encapsulates a log item of the personal information removal task status.
	piremoval.PiRemovalTaskStatus	Object	Server scripts	Encapsulates the status of a personal information removal task. Use piremoval.getTaskStatus(options) to create this object.
Method	piremoval.createTask(options)	piremoval.PiRemovalTask	Server scripts	Creates a personal information removal task.
	piremoval.deleteTask(options)	void	Server scripts	Deletes a personal information removal task.
	piremoval.getTaskStatus(options)	piremoval.PiRemovalTaskStatus	Server scripts	Retrieves the status of a personal information removal task.
	piremoval.loadTask(options)	piremoval.PiRemovalTask	Server scripts	Loads a personal information removal task.

PiRemovalTask Object Members

The following members are called on the [piremoval.PiRemovalTask](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	PiRemovalTask.deleteTask()	void	Server scripts	Deletes the personal information removal task.
	PiRemovalTask.run()	void	Server scripts	Runs the personal information removal task.
	PiRemovalTask.save()	void	Server scripts	Saves the personal information removal task.
Property	PiRemovalTask.fieldIds	string[] (read-only)	Server scripts	Represents the field IDs that are processed by the PI removal task.
	PiRemovalTask.historyOnly	boolean	Server scripts	Indicates whether the PI removal task removes system note information only, not field values or workflow history.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	PiRemovalTask.historyReplacement	string (read-only)	Server scripts	Represents the text used in system notes to replace the original values.
	PiRemovalTask.id	number (read-only)	Server scripts	Represents the ID of the personal information removal task.
	PiRemovalTask.recordIds	number[] (read-only)	Server scripts	Represents the record IDs that are processed by the PI removal task.
	PiRemovalTask.recordType	string (read-only)	Server scripts	Describes the record type updated by the PI removal task.
	PiRemovalTask.status	piremoval.PiRemovalTaskStatus	Server scripts	Describes the status of the submitted personal information removal task.
	PiRemovalTask.workflowIds	number[] (read-only)	Server scripts	Represents the workflow IDs whose history is processed by the PI removal task.

PiRemovalTaskLogItem Object Members

The following members are called on the [piremoval.PiRemovalTaskLogItem](#) object.

Member Type	Name	Return Type/Value Type	Support Script Type	Description
Property	PiRemovalTaskLogItem.exception	string (read-only)	Server scripts	Describes the exception for the log item, including and what caused it.
	PiRemovalTaskLogItem.message	string (read-only)	Server scripts	Describes the message for the log item and an explanation for any errors.
	PiRemovalTaskLogItem.status	string (read-only)	Server scripts	Describes the status of the log item. This property takes its values from task.TaskStatus .
	PiRemovalTaskLogItem.type	string (read-only)	Server scripts	Describes the type of personal information that was removed, one of FieldValue, SystemNote, or Workflow.

PiRemovalTaskStatus Object Members

The following members are called on the [piremoval.PiRemovalTaskStatus](#) object.

Member Type	Name	Return Type/Value Type	Support Script Type	Description
Property	PiRemovalTaskStatus.logList	list	Server scripts	Represents a list of logs for the PI removal task job.

Member Type	Name	Return Type/Value Type	Support Script Type	Description
	PiRemovalTaskStatus.status	string	Server scripts	Describes the status of the submitted personal information removal task.

N/piremoval Module Script Samples

The following script samples demonstrate how to use the features of the N/piremoval module.

Sample 1: Remove customer phone number

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to remove the phone numbers and comments in specific customer records from the record fields (field values), system notes, and workflow history. The removed values are replaced with **removed_value**.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/piremoval'], function(piremoval) {
6   function removePersonalInformation() {
7     var piRemovalTask = piremoval.createTask({
8       recordType: 'customer',
9       recordIds: [11, 19],
10      fieldIds: ['comments', 'phone'],
11      workflowIds: [1],
12      historyOnly: false,
13      historyReplacement: 'removed_value'
14    });
15
16    piRemovalTask.save();
17    var taskId = piRemovalTask.id;
18
19    var piRemovalTaskInProgress = piremoval.loadTask(taskId);
20    piRemovalTaskInProgress.run();
21
22    var status = piremoval.getTaskStatus(taskId);
23  };
24
25  removePersonalInformation();
26 });

```

piremoval.PiRemovalTask

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a task to remove personal information (PI). This object includes lists of the record, field, and workflow IDs to remove personal information from, as well as history replacement information.
---------------------------	---

	Use <code>piremoval.createTask(options)</code> to create a <code>piremoval.PiRemovalTask</code> object, or use <code>piremoval.loadTask(options)</code> to load a PI removal task as a <code>piremoval.PiRemovalTask</code> object. To save the object, use <code>PiRemovalTask.save()</code> . To execute the PI removal, use <code>PiRemovalTask.run()</code> .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/piremoval Module
Methods and Properties	PiRemovalTask Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [11, 19],
6   fieldIds: ['comments', 'phone'],
7   workflowIds: [1],
8   historyOnly: false,
9   historyReplacement: 'removed_value'
10 });
11
12 myPiRemovalTask.save();
13 var myTaskId = myPiRemovalTask.id;
14
15 myPiRemovalTask.run();
16 ...
17 // Add additional code

```

PiRemovalTask.deleteTask()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Deletes the PI Removal task.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members

Since	2019.2
-------	--------

Errors

Error Code	Error Message	Thrown If
UNEXPECTED_ERROR	Cannot delete PiRemoval job that was not saved.	The PI removal job is not saved.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [11, 19],
6   fieldIds: ['comments', 'phone'],
7   workflowIds: [1],
8   historyOnly: false,
9   historyReplacement: 'removed_value'
10 });
11
12 myPiRemovalTask.save();
13 var myTaskId = myPiRemovalTask.id;
14
15 myPiRemovalTask.deleteTask();
16 ...
17 //Add additional code

```

PiRemovalTask.run()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Runs the PI removal task. All validation for the task (for example, ensuring that the specified record IDs are valid) occurs when the task is saved using PiRemovalTask.save() , not when the task is run using PiRemovalTask.run() .
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members

Since	2019.2
--------------	--------

Errors

Error Code	Error Message	Thrown If
UNEXPECTED_ERROR	Cannot run unsaved PiRemoval job.	The PI removal job is not saved.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [11, 19],
6   fieldIds: ['comments', 'phone'],
7   workflowIds: [1],
8   historyOnly: false,
9   historyReplacement: 'removed_value'
10 });
11
12 myPiRemovalTask.save();
13 var my taskId = myPiRemovalTask.id;
14
15 myPiRemovalTask.run();
16 ...
17 // Add additional code

```

PiRemovalTask.save()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Saves the PI removal task. All validation for the task (for example, ensuring that the specified record IDs are valid) occurs when the task is saved using this method.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members
Since	2019.2

Errors

Error Code	Error Message	Thrown If
_1_CANNOT_BE_EMPTY	Record Type cannot be empty.	The record type is not set.
_1_JOB_WAS_NOT_FOUND	Record Type 'type' was not found.	Record type does not exist.
_1_JOB_WAS_NOT_FOUND	Record ID 'ID' was not found.	One of the record IDs does not exist.
_1_JOB_WAS_NOT_FOUND	Field ID 'ID' was not found.	One of the field IDs does not exist.
_1_JOB_WAS_NOT_FOUND	Workflow ID 'ID' was not found.	One of the workflow IDs does not exist.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [11, 19],
6   fieldIds: ['comments', 'phone'],
7   workflowIds: [1],
8   historyOnly: false,
9   historyReplacement: 'removed_value'
10 });
11
12 myPiRemovalTask.save();
13 var my taskId = myPiRemovalTask.id;
14
15 myPiRemovalTask.run();
16 ...
17 // Add additional code

```

PiRemovalTask.fieldIds



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	IDs of the fields whose PI is removed. If no field IDs are entered, no information changes are performed. If the field IDs are null or invalid, the following exception occurs: <ul style="list-style-type: none">■ Wrong parameter type: options.fieldIds is expected as array.
Type	string[] (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var theFieldIds = myPiRemovalTask.fieldIds;
12 ...
13 // Add additional code

```

PiRemovalTask.historyOnly



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the PI removal task removes system note information only, not field values or workflow history. If true, the task removes information from system notes only. If false, the task removes information from system notes, workflow history, and field values. The default value is false.
Type	boolean (read-only)
Module	N/piremoval Module
Parent Object	PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyOnly: true,
8   historyReplacement: 'removed_value'
9 });
10
11 myPiRemovalTask.save();
12 var theHistoryOnly = myPiRemovalTask.historyOnly;

```

```

13 | ...
14 | //Add additional code

```

PiRemovalTask.historyReplacement



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The text used in system notes to replace the original value.
Type	string (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var theHistoryReplacement = myPiRemovalTask.historyReplacement;
12 ...
13 // Add additional code

```

PiRemovalTask.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	ID that uniquely identifies the PI removal task. This ID is assigned to the PI removal task when PiRemovalTask.save() is called to save the task. You cannot specify your own task ID.
Type	number (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members

Since	2019.2
-------	--------

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var theId = myPiRemovalTask.id;
12 ...
13 // Add additional code

```

PiRemovalTask.recordIds

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	ID of records whose PI is removed. If no record IDs are entered, no information changes are performed. If the record IDs are null or invalid, the following exception occurs: <ul style="list-style-type: none">■ Wrong parameter type: options.recordIds is expected as array.
Type	number[] (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members
Since	2019.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95, 107],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9

```

```

10 myPiRemovalTask.save();
11 var theRecordIds = myPiRemovalTask.recordIds;
12 ...
13 // Add additional code

```

PiRemovalTask.recordType

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Type of record whose PI is removed. All records referenced in the piremoval.PiRemovalTask object must be the same type.
Type	string (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: [ 'email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var theRecordType = myPiRemovalTask.recordType;
12 ...
13 // Add additional code

```

PiRemovalTask.status

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status of the PI removal task.
Type	piremoval.PiRemovalTaskStatus
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members

Since	2019.2
-------	--------

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var theStatus = myPiRemovalTask.status;
12 ...
13 // Add additional code

```

PiRemovalTask.workflowIds

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	IDs of workflows where PI is removed from the workflow history. If no workflow IDs are entered, no information changes are performed. If the workflow IDs are null or invalid, the following exception occurs: <ul style="list-style-type: none">■ Wrong parameter type: options.workflowIds is expected as array.
Type	number[] (read-only)
Module	N/piremoval Module
Parent Object	PiRemovalTask
Sibling Object Members	PiRemovalTask Object Members
Since	2019.2

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95, 107],
6   fieldIds: ['email', 'phone'],
7   workflowIds: [1, 7],
8   historyReplacement: 'removed_value'
9 });
10

```

```

11 myPiRemovalTask.save();
12 var theWorkflowIds = myPiRemovalTask.workflowIds;
13 ...
14 // Add additional code

```

piremoval.PiRemovalTaskStatus



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the status of a personal information removal task returned by piremoval.getTaskStatus(options) . Possible status values include: <ul style="list-style-type: none">■ PENDING■ PROCESSING■ COMPLETE■ FAILED For more information, see task.TaskStatus .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/piremoval Module
Methods and Properties	PiRemovalTaskStatus Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myFirstStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16
17 myPiRemovalTask.run();
18
19 var mySecondStatus = piremoval.getTaskStatus({
20   id: myId
21 });
22 ...
23 // Add additional code

```

PiRemovalTaskStatus.logList

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Gets list of logs for the PiRemovalTask job.
Type	piremoval.PiRemovalTaskLogItem[]
Module	N/piremoval Module
Parent Object	PiRemovalTaskStatus
Sibling Object Members	PiRemovalTaskStatus Object Members
Since	2019.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16
17 var theLogList = myStatus.logList;
18 for (var i = 0; i < theLogList.length; i++) {
19   log.debug({
20     title: 'logList value',
21     details: theLogList[i]
22   });
23 }
24 ...
25 // Add additional code

```

PiRemovalTaskStatus.status

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Encapsulates the status of a personal information removal task returned by piremoval.getTaskStatus(options) . Possible status values include: <ul style="list-style-type: none"> ■ PENDING ■ PROCESSING
-----------------------------	--

	<ul style="list-style-type: none"> ■ COMPLETE ■ FAILED <p>For more information, see task.TaskStatus.</p>
Type	string (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTaskStatus
Sibling Object Members	PiRemovalTaskStatus Object Members
Since	2019.2

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myTaskStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16 var theStatus = myTaskStatus.status;
17 ...
18 // Add additional code

```

piremoval.PiRemovalTaskLogItem

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	This object represents log items that are associated with a piremoval.PiRemovalTaskStatus object. The logs are generated separately when a task is created, started, and completed. A piremoval.PiRemovalTaskLogItem object represents a single log entry that you get from the piremoval.PiRemovalTaskStatus object using the PiRemovalTaskStatus.logList property. The log items are sorted by date. The structure of this object is described in PiRemovalTaskLogItem Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/piremoval Module
Methods and Properties	PiRemovalTaskLogItem Object Members

Since	2019.2
-------	--------

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16
17 var theLogList = myStatus.logList;
18 for (var i = 0; i < theLogList.length; i++) {
19   log.debug({
20     title: 'logList value',
21     details: theLogList[i]
22   });
23 }
24 ...
25 // Add additional code

```

PiRemovalTaskLogItem.exception

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Exception message for the log item, typically an unexpected error from NetSuite.
Type	string (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTaskLogItem
Sibling Object Members	PiRemovalTaskLogItem Object Members
Since	2019.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({

```

```

4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16
17 var theLogList = myStatus.logList;
18 for (var i = 0; i < theLogList.length; i++) {
19   var theLogListException = theLogList[i].exception;
20
21   // Do something with the log list exception here
22 }
23 ...
24 // Add additional code

```

PiRemovalTaskLogItem.message



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Log item text message. The message specifies if the record type is not set, or if one of record, field, or workflow IDs do not exist.
Type	string (read-only)
Module	N/piremoval Module
Parent Object	PiRemovalTaskLogItem
Sibling Object Members	PiRemovalTaskLogItem Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16
17 var theLogList = myStatus.logList;

```

```

18 | for (var i = 0; i < theLogList.length; i++) {
19 |   var theLogListMessage = theLogList[i].message;
20 |
21 |   // Do something with the log list message here
22 |
23 | }
24 | ...
25 | // Add additional code

```

PiRemovalTaskLogItem.status



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Encapsulates the status of a log item. Possible status values include: <ul style="list-style-type: none">■ PENDING■ PROCESSING■ COMPLETE■ FAILED For more information, see task.TaskStatus .
Type	string (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTaskLogItem
Sibling Object Members	PiRemovalTaskLogItem Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16
17 var theLogList = myStatus.logList;
18 for (var i = 0; i < theLogList.length; i++) {
19   var theLogListStatus = theLogList[i].status;
20
21   // Do something with the log list status here
22 }
23 ...

```

24 // Add additional code

PiRemovalTaskLogItem.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates the change described by this log item. Possible values include: <ul style="list-style-type: none">■ FieldValue - field value■ SystemNote - system note■ Workflow - workflow history
Type	string (read-only)
Module	N/piremoval Module
Parent Object	piremoval.PiRemovalTaskLogItem
Sibling Object Members	PiRemovalTaskLogItem Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16
17 var theLogList = myStatus.loglist;
18 for (var i = 0; i < theLogList.length; i++) {
19   var theLogListType = theLogList[i].type;
20
21   // Do something with the log list type here
22 }
23 ...
24 // Add additional code

```

piremoval.createTask(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new personal information removal task.
---------------------------	--

Returns	piremoval.PiRemovalTask
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/piremoval Module
Sibling Object Members	N/piremoval Module Members
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fieldIds	number[]	optional	Represents IDs of fields whose personal information is removed.
options.historyOnly	boolean	optional	Indicates whether the PI removal task removes system note information only, not field values or workflow history. If true, the task removes information from system notes only. If false, the task removes information from system notes, workflow history, and field values. The default value is false.
options.historyReplacement	string	optional	Represents the text used in system notes to replace the original values.
options.recordIds	number[]	optional	Represents IDs of records whose personal information is removed.
options.recordType	string	optional	Describes the record type that is updated by the PI removal task.
options.workflowIds	number[]	optional	Represents the workflow IDs whose history is processed by the PI removal task.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95, 107],
6   fieldIds: ['email', 'phone'],
7   workflowIds: [3, 7],
8   historyOnly: true,
9   historyReplacement: 'removed_value'
```

```

10 });
11 myPiRemovalTask.save();
12 var myTaskId = myPiRemovalTask.id;
13
14 myPiRemovalTask.run();
15 ...
16 //Add additional code
17

```

piremoval.deleteTask(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Deletes a personal information removal task.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/piremoval Module
Sibling Object Members	N/piremoval Module Members
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	number	required	Unique identifier of the personal information removal task.

Errors

Error Code	Error Message	Thrown If
UNEXPECTED_ERROR	Cannot delete PI Removal job that was not saved	Job is not saved.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 //Add additional code

```

```

2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95, 107],
6   fieldIds: ['email', 'phone'],
7   workflowIds: [3, 7],
8   historyOnly: true,
9   historyReplacement: 'removed_value'
10 });
11
12 myPiRemovalTask.save();
13 var myTaskId = myPiRemovalTask.id;
14
15 piremoval.deleteTask({
16   id: myTaskId
17 });
18 ...
19 // Add additional code

```

piremoval.getTaskStatus(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves the status of a personal information removal task.
Returns	piremoval.PiRemovalTaskStatus
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/piremoval Module
Sibling Object Members	N/piremoval Module Members
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	number	required	Unique identifier of the personal information removal task.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...

```

```

3  var myPiRemovalTask = piremoval.createTask({
4    recordType: 'customer',
5    recordIds: [95],
6    fieldIds: ['email'],
7    historyReplacement: 'removed_value'
8  });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myFirstStatus = piremoval.getTaskStatus({
14   id: myId
15 });
16
17 myPiRemovalTask.run();
18
19 var mySecondStatus = piremoval.getTaskStatus({
20   id: myId
21 });
22 ...
23 // Add additional code

```

piremoval.loadTask(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves an existing personal information removal task.
Returns	piremoval.PiRemovalTask
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/piremoval Module
Sibling Object Members	N/piremoval Module Members
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	number	required	Unique identifier of the personal information removal task.

Errors

Error Code	Error Message	Thrown If
_1_WAS_NOT_FOUND	PIRemoval job was not found.	Job with the ID provided was not found.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/piremoval Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPiRemovalTask = piremoval.createTask({
4   recordType: 'customer',
5   recordIds: [95],
6   fieldIds: ['email'],
7   historyReplacement: 'removed_value'
8 });
9
10 myPiRemovalTask.save();
11 var myId = myPiRemovalTask.id;
12
13 var myLoadedPiRemovalTask = piremoval.loadTask({
14   id: myId
15 });
16
17 myLoadedPiRemovalTask.run();
18 ...
19 // Add additional code

```

N/plugin Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/plugin module to load custom plug-in implementations. For additional information, see the help topic [Custom Plug-ins](#).



Important: You cannot use the SuiteScript Debugger to debug a script on demand that uses the N/plugin module. You must use deployed debugging. To use deployed debugging, you must complete the steps described in [Adding a Script that Instantiates a Custom Plug-in to NetSuite](#). For the complete process on creating a custom plugin, see the help topic [Custom Plug-in Development](#). For additional information on ad-hoc and deployed debugging, see the help topic [SuiteScript Debugger](#).

- [N/plugin Module Members](#)
- [N/plugin Module Script Samples](#)

N/plugin Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	plugin.findImplementations(options)	string[]	Server-side scripts	Returns the script IDs of custom plug-in type implementations.
Method	plugin.loadImplementation(options)	Object	Server-side scripts	Instantiates an implementation of the custom plug-in type.

N/plugin Module Script Samples

The following script samples demonstrate how to use the features of the N/plugin module.

Sample 1: Find plugin implementations

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows an implementation of the plug-in interface. To test this sample, you need a custom plug-in type with a script ID of customscript_magic_plugin and an interface with a single method, int doTheMagic(int, int).

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType pluginTypeimpl
4 */
5 define(function() {
6     return {
7         doTheMagic: function(operand1, operand2) {
8             return operand1 + operand2;
9         }
10    });
11 });

```

The following Suitelet iterates through all implementations of the custom plug-in type customscript_magic_plugin. For the plug-in to be recognized, the Suitelet script record must specify the plug-in type under Custom Plug-in Types.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4 */
5 define(['N/plugin'], function(plugin) {
6     function onRequest(context) {
7         var impls = plugin.findImplementations({
8             type: 'customscript_magic_plugin'
9         });
10
11         for (var i = 0; i < impls.length; i++) {
12             var pl = plugin.loadImplementation({
13                 type: 'customscript_magic_plugin',
14                 implementation: impls[i]
15             });
16             log.debug('impl ' + impls[i] + ' result = ' + pl.doTheMagic(10, 20));
17         }
18
19         var pl = plugin.loadImplementation({
20             type: 'customscript_magic_plugin'
21         });
22         log.debug('default impl result = ' + pl.doTheMagic(10, 20));
23     }
24
25     return {
26         onRequest: onRequest
27     };
28 });

```

plugin.findImplementations(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description

Returns the script IDs of custom plug-in type implementations.

	Returns an empty list when there is no custom plug-in type with the script ID available for the executing script.
Returns	A string[] containing a list of custom plug-in implementation script IDs.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	
Module	N/config Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	The script ID of the custom plug-in type.	2016.1
options.includeDefault	boolean true false	optional	The default value is true, indicating that the default implementation should be included in the list.	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/plugin Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var impls = plugin.findImplementations({
4   type: 'customscript_sample_plugin'
5 });
6 ...
7 //Add additional code

```

plugin.loadImplementation(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Instantiates an implementation of the custom plugin type. Returns the implementation which is currently selected in the UI (Manage Plug-ins page) when no implementation ID is explicitly provided.
Returns	An Object implementing the custom plug-in type.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	

Module	N/config Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	The script ID of the custom plug-in type.	2016.1
options.implementation	string	optional	The script ID of the custom plug-in implementation.	2016.1

Error Code	Thrown If
UNABLE_TO_FIND_IMPLEMENTATION_1_FOR_PLUGIN_2	Either there is no such implementation of the provided plug-in type, or the plug-in type does not exist.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/plugin Module Script Sample.

```

1 //Add additional code
2 ...
3 var pl = plugin.loadImplementation({
4   type: 'customscript_sample_plugin'
5 });
6 ...
7 //Add additional code

```

N/portlet Module

Note: The content in this help topic pertains to SuiteScript 2.0.

Load the portlet module to resize or refresh a form portlet. See the help topic [SuiteScript 2.0 Portlet Script Type](#).

- [N/portlet Module Members](#)
- [N/portlet Module Script Sample](#)

N/portlet Module Members

Member Type	Name	Return Type	Supported Script Types	Description
Method	portlet.resize	void	Client scripts	Resizes a form portlet immediately.

Member Type	Name	Return Type	Supported Script Types	Description
	portlet.refresh	void	Client scripts	Refreshes a form portlet immediately.

N/portlet Module Script Sample

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to create a form portlet that allows users to adjust its height and width. It creates two text fields representing the height and width of the portlet, measured in pixels. It also creates a button that runs the resize function to adjust the height and width of the portlet based on the values of the text fields.

This sample also shows how to create a button that uses the refresh function. When pressed, the portlet is updated to show the current date.

For more information about how a portlet is displayed on the NetSuite dashboard, see the help topic [SuiteScript 2.0 Portlet Script Type](#).

```

1  /**
2   * @NApiVersion 2.0
3   * @NScriptType portlet
4   * @NScriptPortletType form
5   */
6
7 define([], function() {
8     function render(context) {
9         var portletObj = context.portlet;
10        portletObj.title = 'Test Form Portlet';
11        setComponentsForResize();
12        setComponentsForRefresh();
13
14        function setComponentsForResize() {
15            var DEFAULT_HEIGHT = '50';
16            var DEFAULT_WIDTH = '50';
17            var inlineHTMLField = portletObj.addField({
18                id: 'divfield',
19                type: 'inlinehtml',
20                label: 'Test inline HTML'
21            });
22            inlineHTMLField.defaultValue = "<div id='divfield_elem' style='border: 1px dotted red; height: " + DEFAULT_HEIGHT +
"px; width: " + DEFAULT_WIDTH + "px'></div>";
23            inlineHTMLField.updateLayoutType({
24                layoutType: 'normal'
25            });
26            inlineHTMLField.updateBreakType({
27                breakType: 'startcol'
28            });
29            var resizeHeight = portletObj.addField({
30                id: 'resize_height',
31                type: 'text',
32                label: 'Resize Height'
33            });
34            resizeHeight.defaultValue = DEFAULT_HEIGHT;
35            var resizeWidth = portletObj.addField({
36                id: 'resize_width',
37                type: 'text',
38                label: 'Resize Width'
39            });
40            resizeWidth.defaultValue = DEFAULT_WIDTH;

```

```

41     var resizeLink = portletObj.addField({
42         id: 'resize_link',
43         type: 'inlinehtml',
44         label: 'Resize link'
45     });
46     resizeLink.defaultValue = resizeLink.defaultValue = "<a id='resize_link' onclick=\"require(['SuiteScripts/portletApiTestHelper'], function(portletApiTestHelper) {portletApiTestHelper.changeSizeAndResizePortlet(); }) \\" href=\"#'>Resize</a><br>";
47 }
48 function setComponentsForRefresh() {
49     var textField = portletObj.addField({
50         id: 'refresh_output',
51         type: 'text',
52         label: 'Date.now().toString()'
53     });
54     textField.defaultValue = Date.now().toString();
55     var refreshLink = portletObj.addField({
56         id: 'refresh_link',
57         type: 'inlinehtml',
58         label: 'Refresh link'
59     });
60     refreshLink.defaultValue = "<a id='refresh_link' onclick=\"require(['SuiteScripts/portletApiTestHelper'],
61     function(portletApiTestHelper) {portletApiTestHelper.refreshPortlet(); }) \\" href=\"#'>Refresh</a>\"";
62 }
63 return {
64     render: render
65 };
66 })
67
68 // portletApiTestHelper.js
69 define(['N/portlet'], function(portlet) {
70     function refreshPortlet() {
71         portlet.refresh();
72     }
73
74     function resizePortlet() {
75         var div = document.getElementById('divfield_elem');
76         var newHeight = parseInt(document.getElementById('resize_height').value);
77         var newWidth = parseInt(document.getElementById('resize_width').value);
78         div.style.height = newHeight + 'px';
79         div.style.width = newWidth + 'px';
80         portlet.resize();
81     }
82     return {
83         refreshPortlet: refreshPortlet,
84         resizePortlet: resizePortlet
85     };
86 })

```

portlet.resize



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Resizes a form portlet type immediately.
Returns	Void
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/portlet Module
Since	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/portlet Module Script Sample](#)

```

1 //Add additional code
2 ...
3 portlet.resize();
4 ...
5 //Add additional code

```

portlet.refresh



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Refreshes a form portlet type immediately.
Returns	Void
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/portlet Module
Since	2016.1

Syntax

The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/portlet Module Script Sample](#)

```

1 ...
2 portlet.refresh();
3 ...

```

N/query Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/query module to create and run queries using the SuiteAnalytics Workbook query engine. For more information about SuiteAnalytics Workbook, see the help topic [SuiteAnalytics Workbook Overview](#).

Using the query module, you can:

- Use multilevel joins to create queries using field data from multiple record types.
- Create conditions (filters) using AND, OR, and NOT logic, as well as formulas and relative dates.
- Sort query results based on the values of multiple columns.
- Load and delete existing saved queries that were created using the SuiteAnalytics Workbook interface.

- View paged query results.
- Use promises for asynchronous execution.
- Convert query objects to SuiteQL queries and run arbitrary SuiteQL queries.

For more information about creating scripts using the N/query module, see the following help topics:

- [Scripting with the N/query Module](#)
- [Formulas in the N/query Module](#)
- [Relative Dates in the N/query Module](#)
- [SuiteQL in the N/query Module](#)



Important: As you use the N/query module, keep the following considerations in mind:

- The N/query module lets you create and run queries using the SuiteAnalytics Workbook query engine. You can use the N/query module to load and delete existing queries, but you cannot save queries. You can save queries using the SuiteAnalytics Workbook interface. For more information, see the help topic [Navigating SuiteAnalytics Workbook](#).
- The N/query module uses a different data source than the N/search module. If you need to determine the ID of a record type or field (for example, to use as a query filter or result column), you must obtain this value from the Analytics Browser, not the SuiteScript Records Browser. The Analytics Browser lists every record type and field that is currently available in SuiteAnalytics Workbook and the N/query module. For more information, see the help topic [Analytics Browser](#).
- The N/query module supports the same record types that are supported in the SuiteAnalytics Workbook interface. For more information, see the help topic [Finding Record Types in the Analytics Browser](#).

In This Help Topic

- [N/query Module Members](#)
- [Column Object Members](#)
- [Component Object Members](#)
- [Condition Object Members](#)
- [Page Object Members](#)
- [PagedData Object Members](#)
- [PageRange Object Members](#)
- [Query Object Members](#)
- [RelativeDate Object Members](#)
- [Result Object Members](#)
- [ResultSet Object Members](#)
- [Sort Object Members](#)
- [SuiteQL Object Members](#)
- [N/query Module Script Samples](#)

N/query Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	query.Column	Object	Client and server scripts	<p>The field types (query result columns) that are displayed from the query results.</p> <p>Use Query.createColumn(options) or Component.createColumn(options) to create this object.</p>
	query.Component	Object	Client and server scripts	<p>One component of the query definition. The query definition always contains at least one component that encapsulates the initial query type. Queries with joins contain multiple components that encapsulate the join relationships.</p> <p>The initial component (Query.root) is automatically created with the query definition (query.Query). Use Query.autoJoin(options) or Component.autoJoin(options) to create subsequent components.</p>
	query.Condition	Object	Client and server scripts	<p>A condition. A condition narrows the query results.</p> <p>Use Query.createCondition(options) or Component.createCondition(options) to create this object.</p>
	query.Page	Object	Client and server scripts	One page of the paged query results.
	query.PagedData	Object	Client and server scripts	A set of paged query results. This object also contains information about the set of paged results it encapsulates.
	query.PageRange	Object	Client and server scripts	A range of pages from the paged query results.
	query.Period	Object	Client and server scripts	A period of time to use in query conditions.
	query.RelativeDate	Object	Client and server scripts	A relative date to use in query conditions.
	query.Result	Object	Client and server scripts	A single row of the query result set.
	query.ResultSet	Object	Client and server scripts	The set of results returned by the query.
Object	query.Query	Object	Client and server scripts	<p>The query definition. Use query.create(options) or query.load(options) to create this object.</p> <p>The creation of this object is the first step in creating a query with the N/query Module.</p>
	query.Sort	Object	Client and server scripts	<p>A sort that is placed on a particular query result column.</p> <p>Use Query.createSort(options) or Component.createSort(options) to create this object.</p>

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	query.SuiteQL	Object	Client and server scripts	A SuiteQL query. Use Query.toSuiteQL() to create this object.
Method	query.create(options)	query.Query	Client and server scripts	Creates the query definition. The execution of this method is the first step in creating a query with the N/query Module.
	query.createPeriod(options)	query.Period	Client and server scripts	Creates a query.Period object that represents a period of time.
	query.createRelativeDate(options)	query.RelativeDate	Client and server scripts	Creates a query.RelativeDate object that represents a date relative to the current date.
	query.delete(options)	void	Client and server scripts	Deletes an existing query that was created using the SuiteAnalytics Workbook UI. The deleted query is no longer available and cannot be modified or executed.
	query.listTables(options)	Array<Object>	Client and server scripts	Lists the table view objects that are included in a workbook in SuiteAnalytics Workbook.
	query.load(options)	query.Query	Client and server scripts	Loads an existing query that was created using the SuiteAnalytics Workbook UI. The loaded query can be modified (for example, by setting additional property values), joined with other query types, and executed in the same way as queries created using query.create(options) .
	query.load.promise(options)	query.Query	Client scripts	Asynchronously loads an existing query that was created using the SuiteAnalytics Workbook UI.
	query.runSuiteQL(options)	query.ResultSet	Client and server scripts	Runs an arbitrary SuiteQL query.
Enum	query.runSuiteQLPaged(options)	query.PagedData	Client and server scripts	Runs an arbitrary SuiteQL query as a paged query.
	query.Aggregate	enum	Client and server scripts	Holds the string values for aggregate functions supported with the N/query Module . This enum is used to pass the aggregate function argument to Component.createColumn(options) , Component.createCondition(options) , Query.createColumn(options) , and Query.createCondition(options) .
	query.DateId	enum	Client and server scripts	Holds the string values for supported date codes in relative dates. This enum is used to pass the date ID argument to query.createRelativeDate(options) .
	query.FieldContext	enum	Client and server scripts	Holds the string values for the field context to use when creating a column.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				This enum is used to pass the context argument to Query.createColumn(options) and Component.createColumn(options) .
	query.Operator	enum	Client and server scripts	Holds the string values for operators supported with the N/query Module . This enum is used to pass the operator argument to Query.createCondition(options) and Component.createCondition(options) .
	query.PeriodAdjustment	enum	Client and server scripts	Holds the string values for adjustment types for a period. This enum is used to pass the adjustment argument to query.createPeriod(options) .
	query.PeriodCode	enum	Client and server scripts	Holds the string values for period codes for a period. This enum is used to pass the code argument to query.createPeriod(options) .
	query.PeriodType	enum	Client and server scripts	Holds the string values for period types for a period. This enum is used to pass the type argument to query.createPeriod(options) .
	query.RelativeDateRange	enum	Client and server scripts	Holds query.RelativeDate object values for supported date ranges in relative dates. This enum is used to pass the values argument to Query.createCondition(options) and Component.createCondition(options) .
	query.ReturnType	enum	Client and server scripts	Holds the string values for the formula return types supported with the N/query Module . This enum is used to pass the formula return type argument to Query.createColumn(options) , Component.createColumn(options) , Query.createCondition(options) , and Component.createCondition(options) .
	query.SortLocale	enum	Client and server scripts	Holds the string values for sort locales supported with the N/query Module . This enum is used to pass the sort locale argument to Query.createSort(options) and Component.createSort(options) .
	query.Type	enum	Client and server scripts	Holds the string values for supported query types used in the query definition. This enum is used to pass the initial query type argument to query.create(options) .

Column Object Members

The following members are called on the [query.Column](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Property	Column.aggregate	string (read-only)	Client and server scripts	An aggregate function that is performed on the query result column. An aggregate function performs a calculation on the column values and returns a single value.
	Column.alias	string (read-only)	Client and server scripts	An alias for this column. An alias is an alternate name for a column, and the alias is used in mapped results.
	Column.component	query.Component (read-only)	Client and server scripts	A reference to the query.Component object to which this query result column belongs.
	Column.context	Object (read-only)	Client and server scripts	The field context for values in the query result column. The field context determines how field values are displayed in the column.
	Column.fieldId	string (read-only)	Client and server scripts	The name of the query result column. This property and the Column.formula property cannot be set at the same time.
	Column.formula	string (read-only)	Client and server scripts	The formula used to create the query result column. This property and the Column.fieldId property cannot be set at the same time.
	Column.groupBy	boolean (read-only)	Client and server scripts	Whether the query results are grouped by this query result column.
	Column.label	string (read-only)	Client and server scripts	The label for the column.
	Column.type	string (read-only)	Client and server scripts	The return type of the formula used to create the query result column.

Component Object Members

The following members are called on the [query.Component](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	Component.autoJoin(options)	query.Component	Client and server scripts	<p>Creates a join relationship.</p> <p>After you create the initial query definition, use Query.autoJoin(options) to create your</p>

Member Type	Name	Return Type/ Value Type	Supported Script Types	Description
				first join. Then use this method to create each subsequent join. This method selects the correct join type automatically based on the record types that are being joined.
	Component.createColumn(options)	query.Column	Client and server scripts	Creates a query result column based on the component. Use this method to create columns based on the join relationships created with Query.autoJoin(options) and Component.autoJoin(options) .
	Component.createCondition(options)	query.Condition	Client and server scripts	Creates a condition (filter column) based on the component. Use this method to create conditions based on the join relationships created with Query.autoJoin(options) and Component.autoJoin(options) .
	Component.createSort(options)	query.Sort	Client and server scripts	Creates a sort based on the component. Use this method to create sorts based on the join relationships created with Query.autoJoin(options) and Component.autoJoin(options) .
	Component.join(options)	query.Component	Client and server scripts	Creates a join relationship. This method is an alias to Component.autoJoin(options) . After you create the initial query definition, use Query.autoJoin(options) to create your first join. Then use this method, or Component.autoJoin(options) , to create each subsequent join.
	Component.joinFrom(options)	query.Component	Client and server scripts	Creates an explicit directional join relationship from another component to this component (an inverse join). This method sets the Component.source property on the returned query.Component object. After you create the initial query definition, use this method to create explicit directional joins from other components to this component.
	Component.joinTo(options)	query.Component	Client and server scripts	Creates an explicit directional join relationship to another component from this component (a polymorphic join). You can use this method to specify the target of the join when a field can join multiple query types. This method sets the Component.target property on the returned query.Component object. After you create the initial query definition, use this method to create explicit directional joins to other components from this component.
Property	Component.child	Object (read-only)	Client and server scripts	The child components of the component. This property holds an object of key/value pairs. Each key is the name of a child

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
				component. Each value is the corresponding child query.Component object.
	Component.parent	string (read-only)	Client and server scripts	The parent query.Component object of the component.
	Component.source	string (read-only)	Client and server scripts	The source query type of the component. The value of this property is set when Component.joinFrom(options) is called to perform an explicit directional join from another component.
	Component.target	string (read-only)	Client and server scripts	The target query type of the component. The value of this property is set when Component.joinTo(options) is called to perform an explicit directional join to another component.
	Component.type	string (read-only)	Client and server scripts	The query type of the component.

Condition Object Members

The following members are called on the [query.Condition](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Property	Condition.aggregate	string (read-only)	Client and server scripts	An aggregate function that is performed on the condition. An aggregate function performs a calculation on the condition values and returns a single value.
	Condition.children	query.Condition [] (read-only)	Client and server scripts	An array of child conditions used to create the parent condition.
	Condition.component	query.Component (read-only)	Client and server scripts	A reference to the query.Component object to which this condition belongs.
	Condition.fieldId	string (read-only)	Client and server scripts	The name of the field that is used in the condition.
	Condition.formula	string (read-only)	Client and server scripts	The formula used to create the condition.
	Condition.operator	string (read-only)	Client and server scripts	The name of the operator used to create the condition.
	Condition.type	string (read-only)	Client and server scripts	The return type of the formula used to create the condition.
	Condition.values	string number boolean Array<string number boolean> (read-only)	Client and server scripts	An array of values used by an operator to create the condition.

Page Object Members

The following members are called on the [query.Page](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Property	Page.data	query.ResultSet (read-only)	Client and server scripts	The query results contained in this page.
	Page.isFirst	boolean (read-only)	Client and server scripts	Whether this page is the first of the paged query results.
	Page.isLast	boolean (read-only)	Client and server scripts	Whether this page is the last of the paged query results.
	Page.pagedData	query.PagedData (read-only)	Client and server scripts	The set of paged query results that this page is from.
	Page.pageRange	query.PageRange (read-only)	Client and server scripts	The range of query results for this page.

PagedData Object Members

The following members are called on the [query.PagedData](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	PagedData.iterator()	Iterator object	Client and server scripts	Standard SuiteScript 2.0 object for iterating through results.
Property	PagedData.count	number (read-only)	Client and server scripts	The total number of paged query results.
	PagedData.pageRanges	query.PageRange[]	Client and server scripts	An array of page ranges for the set of paged query results.
	PagedData.pageSize	number (read-only)	Client and server scripts	The number of query result rows per page.

PageRange Object Members

The following members are called on the [query.PageRange](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Property	PageRange.index	number (read-only)	Client and server scripts	The array index for this page range.
	PageRange.size	number (read-only)	Client and server scripts	The number of query result rows in this page range.

Period Object Members

The following members are called on the [query.Period](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Property	Period.adjustment	string (read-only)	Client and server scripts	The adjustment of the period. This property uses values from the query.PeriodAdjustment enum.
	Period.code	string (read-only)	Client and server scripts	The code of the period. This property uses values from the query.PeriodCode enum.
	Period.type	string (read-only)	Client and server scripts	The type of the period. This property uses values from the query.PeriodType enum.

Query Object Members

The following members are called on the [query.Query](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	Query.and()	query.Condition	Client and server scripts	Creates a new condition (a query.Condition object) that corresponds to a logical conjunction (AND) of the arguments passed to the method. The arguments must be one or more query.Condition objects.
	Query.autoJoin(options)	query.Component	Client and server scripts	Creates a join relationship. After you create the initial query definition, use this method to create your first join or subsequent joins from the root component of the query. This method selects the correct join type automatically based on the record types that are being joined.
	Query.createColumn(options)	query.Column	Client and server scripts	Creates a query result column based on the query.Query object. Use this method to create columns on the initial query definition created with query.create(options) .
	Query.createCondition(options)	query.Condition	Client and server scripts	Creates a condition (filter column) based on the query.Query object. Use this method to create conditions on the initial query definition created with query.create(options) .
	Query.createSort(options)	query.Sort	Client and server scripts	Creates a sort based on the query.Query object. The query.Sort object describes a sort that is placed on a particular query result column or condition.

Member Type	Name	Return Type/ Value Type	Supported Script Types	Description
	Query.join(options)	query.Component	Client and server scripts	<p>Creates a join relationship. This method is an alias to Query.autoJoin(options).</p> <p>After you create the initial query definition, use this method, or Query.autoJoin(options), to create your first join.</p>
	Query.joinFrom(options)	query.Component	Client and server scripts	<p>Creates an explicit directional join relationship from another component to the root component of the search definition (an inverse join). This method sets the Component.source property on the returned query.Component object.</p> <p>After you create the initial query definition, use this method to create your first join as an explicit directional join from another component to this component.</p>
	Query.joinTo(options)	query.Component	Client and server scripts	<p>Creates an explicit directional join relationship to another component from this component (a polymorphic join). You can use this method to specify the target of the join when a field can join multiple query types. This method sets the Component.target property on the returned query.Component object.</p> <p>After you create the initial query definition, use this method to create your first join as an explicit directional join to another component from this component.</p>
	Query.not()	query.Condition	Client and server scripts	Creates a new condition (a query.Condition object) that corresponds to a logical negation (NOT) of the argument passed to the method. The argument must be a query.Condition object.
	Query.or()	query.Condition	Client and server scripts	Creates a new condition (a query.Condition object) that corresponds to a logical disjunction (OR) of the arguments passed to the method. The arguments must be one or more query.Condition objects.
	Query.run()	query.ResultSet	Client and server scripts	Executes the query and returns the query result set.
	Query.run.promise()	query.ResultSet	Client scripts	Executes the query asynchronously and returns the query result set.
	Query.runPaged()	query.PagedData	Client and server scripts	Executes the query and returns a set of paged results.
	Query.runPaged.promise()	query.PagedData	Client scripts	Executes the query asynchronously and returns a set of paged results.
	Query.toSuiteQL()	query.SuiteQL	Client and server scripts	Converts this query.Query object to its corresponding SuiteQL representation.
Property	Query.child	Object (read-only)	Client and server scripts	A reference to children of the root component of the query definition. The value of this property is an object of key/value pairs. Each key is the name of a child component. Each respective value is the corresponding query.Component object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
	Query.columns	query.Column[]	Client and server scripts	An array of query result columns returned from the query. Before you execute the query, you must assign all created columns as array values to this property.
	Query.condition	query.Condition object	Client and server scripts	The parent condition that narrows the query results. Before you execute the query, you must assign your simple or complex conditions to this property.
	Query.id	number (read-only)	Client and server scripts	The ID of the query definition. This property has a value only for existing queries that are loaded using <code>query.load(options)</code> . If you create a query using <code>query.create(options)</code> but do not save it, this property is null.
	Query.name	string (read-only)	Client and server scripts	The name of the query definition. This property has a value only for existing queries that are loaded using <code>query.load(options)</code> . If you create a query using <code>query.create(options)</code> but do not save it, this property is null.
	Query.root	query.Component (read-only)	Client and server scripts	The root component of the query definition.
	Query.sort	query.Column[] (read-only)	Client and server scripts	An array of query result columns used for sorting.
	Query.type	string (read-only)	Client and server scripts	The query type of the initial query definition.

RelativeDate Object Members

The following members are called on the `query.RelativeDate` object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Property	RelativeDate.dateId	string (read-only)	Client and server scripts	The ID of the relative date.
	RelativeDate.end	Object (read-only)	Client and server scripts	The end point of the relative date.
	RelativeDate.interval	Object (read-only)	Client and server scripts	The interval of the relative date (from the <code>RelativeDate.start</code> point to the <code>RelativeDate.end</code> point).
	RelativeDate.isRange	boolean (read-only)	Client and server scripts	Whether the relative date represents a range of dates or a specific moment in time.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
	RelativeDate.start	Object (read-only)	Client and server scripts	The start point of the relative date.
	RelativeDate.value	number (read-only)	Client and server scripts	The value of the relative date.

Result Object Members

The following members are called on the [query.Result](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	Result.asMap()	Object	Client and server scripts	The query result as a mapped result.
Property	Result.values	Array<string number boolean null> (read-only)	Client and server scripts	The result values.

ResultSet Object Members

The following members are called on the [query.ResultSet](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	ResultSet.asMappedResults()	Object[]	Client and server scripts	Returns a query result set as an array of mapped results.
	ResultSet.iterator()	Iterator object	Client and server scripts	Standard SuiteScript 2.0 object for iterating through results.
Property	ResultSet.columns	query.Column[] (read-only)	Client and server scripts	An array of query result column references.
	ResultSet.results	query.Result[] (read-only)	Client and server scripts	An array of query.Result objects.
	ResultSet.types	string[] (read-only)	Client and server scripts	An array of the return types for ResultSet.results .

Sort Object Members

The following members are called on the [query.Sort](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Property	Sort.ascending	boolean	Client and server scripts	Whether the sort direction is ascending.
	Sort.caseSensitive	boolean	Client and server scripts	Whether the sort is case sensitive.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
				If a sort is case sensitive (and the sort direction is ascending), rows with column values that start with uppercase letters are listed before rows with column values that start with lowercase letters. If a sort is not case sensitive, uppercase and lowercase letters are treated the same.
	Sort.column	query.Column (read-only)	Client and server scripts	The query result column that the query results are sorted by.
	Sort.locale	string	Client and server scripts	The locale to use for the sort. A locale represents a combination of language and region, and it can affect how certain values (such as strings) are sorted.
	Sort.nullsLast	boolean	Client and server scripts	Whether query results with null values are listed at the end of the query results.

SuiteQL Object Members

The following members are called on the [query.SuiteQL](#) object.

Member Type	Name	Return Type/Value Type	Supported Script Types	Description
Method	SuiteQL.run()	query.ResultSet	Client and server scripts	Runs the SuiteQL query and returns the query results.
	SuiteQL.runPaged (options)	query.PagedData	Client and server scripts	Runs the SuiteQL query as a paged query and returns the paged query results.
Property	SuiteQL.columns	query.Column[]	Client and server scripts	Describes the result columns to be returned from the query.
	SuiteQL.params	Array<string number boolean> (read-only)	Client and server scripts	Contains the parameters for the query.
	SuiteQL.query	string (read-only)	Client and server scripts	Holds the string representation of the query.
	SuiteQL.type	string (read-only)	Client and server scripts	Describes the type of the query. This property uses values from the query.Type enum.

N/query Module Script Samples

The following script samples demonstrate how to use the features of the N/query module:

- Sample 1: Query for customer records using joins
- Sample 2: Query for transaction records using joins
- Sample 3: Convert a query to SuiteQL and run it
- Sample 4: Run an arbitrary SuiteQL query

Sample 1: Query for customer records using joins

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a query for customer records, joins the query with two other query types, and runs the query:

```

1  /**
2  * @NApiVersion 2.x
3  */
4
5  require(['N/query'],
6  function(query) {
7
8      // Create a query definition for customer records
9      var myCustomerQuery = query.create({
10          type: query.Type.CUSTOMER
11      });
12
13      // Join the original query definition based on the salesrep field. In a customer
14      // record, the salesrep field contains a reference to an employee record. When you
15      // join based on this field, you are joining the query definition with the employee
16      // query type, and you can access the fields of the joined employee record in
17      // your query.
18      var mySalesRepJoin = myCustomerQuery.autoJoin({
19          fieldId: 'salesrep'
20      });
21
22      // Join the joined query definition based on the location field. In an employee
23      // record, the location field contains a reference to a location record.
24      var myLocationJoin = mySalesRepJoin.autoJoin({
25          fieldId: 'location'
26      });
27
28      // Create conditions for the query
29      var firstCondition = myCustomerQuery.createCondition({
30          fieldId: 'id',
31          operator: query.Operator.EQUAL,
32          values: 107
33      });
34      var secondCondition = myCustomerQuery.createCondition({
35          fieldId: 'id',
36          operator: query.Operator.EQUAL,
37          values: 2647
38      });
39      var thirdCondition = mySalesRepJoin.createCondition({
40          fieldId: 'email',
41          operator: query.Operator.START_WITH_NOT,
42          values: 'foo'
43      });
44
45      // Combine conditions using and() and or() operator methods. In this example,
46      // the combined condition states that the id field of the customer record must
47      // have a value of either 107 or 2647, and the email field of the employee
48      // record (the record that is referenced in the salesrep field of the customer
49      // record) must not start with 'foo'.
50      myCustomerQuery.condition = myCustomerQuery.and(
51          thirdCondition, myCustomerQuery.or(firstCondition, secondCondition)
52      );

```

```

53     // Create query columns
54     myCustomerQuery.columns = [
55       myCustomerQuery.createColumn({
56         fieldId: 'entityid'
57       }),
58       myCustomerQuery.createColumn({
59         fieldId: 'id'
60       }),
61       mySalesRepJoin.createColumn({
62         fieldId: 'entityid'
63       }),
64       mySalesRepJoin.createColumn({
65         fieldId: 'email'
66       }),
67       mySalesRepJoin.createColumn({
68         fieldId: 'hiredate'
69       }),
70       myLocationJoin.createColumn({
71         fieldId: 'name'
72       })
73     ];
74
75
76     // Sort the query results based on query columns
77     myCustomerQuery.sort = [
78       myCustomerQuery.createSort({
79         column: myCustomerQuery.columns[3]
80       }),
81       myCustomerQuery.createSort({
82         column: myCustomerQuery.columns[0],
83         ascending: false
84       })
85     ];
86
87     // Run the query
88     var resultSet = myCustomerQuery.run();
89
90     // Retrieve and log the results
91     var results = resultSet.results;
92     for (var i = results.length - 1; i >= 0; i--)
93       log.debug(results[i].values);
94     log.debug(resultSet.types);
95   });

```

Sample 2: Query for transaction records using joins

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a query for transaction records, joins the query with another query type, and runs the query as a paged query:

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/query'],
6   function(query) {
7
8     // Create a query definition for transaction records
9     var myTransactionQuery = query.create({
10       type: query.Type.TRANSACTION
11     });
12
13     //Join the original query definition based on the employee field. In a transaction
14     //record, the employee field contains a reference to an employee record. When you

```

```

15 // join based on this field, you are joining the query definition with the employee
16 // query type, and you can access the fields of the joined employee record in
17 // your query.
18 var myEmployeeJoin = myTransactionQuery.autoJoin({
19   fieldId: 'employee'
20 });
21
22 // Create a query column
23 myTransactionQuery.columns = [
24   myEmployeeJoin.createColumn({
25     fieldId: 'subsidiary'
26   })
27 ];
28
29 // Sort the query results based on a query column
30 myTransactionQuery.sort = [
31   myTransactionQuery.createSort({
32     column: myTransactionQuery.columns[0],
33     ascending: false
34   })
35 ];
36
37 // Run the query as a paged query with 10 results per page
38 var results = myTransactionQuery.runPaged({
39   pageSize: 10
40 });
41
42 log.debug(results.pageRanges.length);
43 log.debug(results.count);
44
45 // Retrieve the query results using an iterator
46 var iterator = results.iterator();
47 iterator.each(function(result) {
48   var page = result.value;
49   log.debug(page.pageRange.size);
50   return true;
51 });
52
53 // Alternatively, retrieve the query results by looping through
54 // each result
55 for (var i = 0; i < results.pageRanges.length; i++) {
56   var page = results.fetch(i);
57   log.debug(page.pageRange.size);
58 }
59 });

```

Sample 3: Convert a query to SuiteQL and run it

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a query for customer records, converts it to its SuiteQL representation, and runs it:

```

1 /**
2  * @NApiVersion 2.x
3 */
4 require(['N/query'], function(query) {
5   var myCustomerQuery = query.create({
6     type: query.Type.CUSTOMER
7   });
8
9   myCustomerQuery.columns = [
10     myCustomerQuery.createColumn({
11       fieldId: 'entityid'
12     }),

```

```

13     myCustomerQuery.createColumn({
14         fieldId: 'email'
15     });
16 };
17
18 myCustomerQuery.condition = myCustomerQuery.createCondition({
19     fieldId: 'isperson',
20     operator: query.Operator.IS,
21     values: [true]
22 });
23
24 var mySQLCustomerQuery = myCustomerQuery.toSuiteQL();
25
26 var results = mySQLCustomerQuery.run();
27 });

```

Sample 4: Run an arbitrary SuiteQL query

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample constructs a SuiteQL query string, runs the query as a paged query, and iterates over the results:

```

1 /**
2 * @NApiVersion 2.x
3 */
4 require(['N/query'], function(query) {
5     var sql =
6         "SELECT " +
7         " scriptDeployment.primarykey, scriptexecutioncontextmap.executioncontext " +
8         " FROM " +
9         " scriptDeployment, scriptexecutioncontextmap " +
10        " WHERE " +
11        " scriptexecutioncontextmap.scriptrecord = scriptDeployment.primarykey " +
12        " AND " +
13        " scriptexecutioncontextmap.executioncontext IN ('WEBSTORE', 'WEBAPPLICATION')";
14
15     var resultIterator = query.runSuiteQLPaged({
16         query: sql,
17         pageSize: 10
18     }).iterator();
19
20     resultIterator.each(function(page) {
21         var pageIterator = page.value.data.iterator();
22         pageIterator.each(function(row) {
23             log.debug('ID: ' + row.value.getValue(0), 'Context: ' + row.value.getValue(1));
24             return true;
25         });
26         return true;
27     });
28 });

```

Sample 5: Query for a custom field

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a query for a custom field, custrecord_my_custom_field, and obtains the internal ID of the field.

```

1  /*
2  * @NApiVersion 2.x
3  */
4  require(['N/query'], function(query) {
5      var customFieldIdQuery = query.create({
6          type: query.Type.CUSTOM_FIELD
7      });
8      customFieldIdQuery.columns = [
9          customFieldIdQuery.createColumn({
10              fieldId: 'internalid'
11          })
12      ];
13      customFieldIdQuery.condition = customFieldIdQuery.createCondition({
14          fieldId: 'scriptid',
15          operator: query.Operator.IS,
16          values: 'custrecord_my_custom_field'
17      });
18
19      var results = customFieldIdQuery.run().asMappedResults();
20      var customFieldInternalId = results[0].internalid;
21      log.debug(customFieldInternalId);
22  });

```

Scripting with the N/query Module

The N/query module lets you create and run queries using the SuiteAnalytics Workbook query engine. Before you start creating your queries, you should be familiar with the module objects and how to use them, as well as some of the terminology used in the N/query module. You can also take a look at a script walkthrough that explains how to create queries using different approaches.

- [N/query Module Objects](#)
- [N/query Module Terminology](#)

N/query Module Objects

The N/query module includes the following objects:

- [Query and Component Objects](#)
- [Condition Object](#)
- [RelativeDate Object](#)
- [Column Object](#)
- [Sort Object](#)
- [ResultSet and Result Objects](#)
- [Page, PagedData, and PageRange Objects](#)

Query and Component Objects

The [query.Query](#) object and the [query.Component](#) object are the primary building blocks for a query created with the N/query module. Each query creates one [query.Query](#) object and one or more [query.Component](#) objects. The [query.Query](#) object encapsulates the query definition, and the [query.Component](#) object encapsulates one component of the query definition.

To create a query with the N/query module:

1. Use the [query.create\(options\)](#) method to create your initial query definition (the [query.Query](#) object). The initial query definition uses one search type. For available search types, see [query.Type](#).

2. After you create the initial query definition, use `Query.autoJoin(options)`, `Query.joinFrom(options)`, or `Query.joinTo(options)` to create your first join.
3. Use any of the following methods to create subsequent joins:
 - `Query.autoJoin(options)`
 - `Query.joinFrom(options)`
 - `Query.joinTo(options)`
 - `Component.autoJoin(options)`
 - `Component.joinFrom(options)`
 - `Component.joinTo(options)`

The query definition always contains at least one `query.Component` object. Each new component is created as a child of the previous component, and all components exist as children of the query definition. You can think of a component as a building block; each new component builds on the previous component created. The last component created encapsulates the relationship between it and all of its parent components.

Queries with joins contain multiple components. The query definition contains a child `query.Component` object for each of the following:

- **The initial query definition:** The initial `query.Component` object is called the root component. It encapsulates the initial search type passed to `query.create(options)`. The root component is automatically created with the initial query definition and is a child to the `query.Query` object. The `Query.root` property contains a reference to the root component.
- **The first join:** The second `query.Component` object is created with `Query.autoJoin(options)`, `Query.joinFrom(options)`, or `Query.joinTo(options)`. It encapsulates the relationship between the initial query definition and the second search type. This relationship is determined by the join ID passed to these methods, as well as whether `Query.joinFrom(options)` or `Query.joinTo(options)` was used to create an explicit directional join. The second `query.Component` object is a child to the root component.
- **Each subsequent join:** The third `query.Component` object is created with `Component.autoJoin(options)`, `Component.joinFrom(options)`, or `Component.joinTo(options)`. All subsequent joins are also created using these methods. Each of these `query.Component` objects encapsulates the relationship between all previous search types and the new search type. This relationship is determined by the join ID passed to these methods, as well as whether `Component.joinFrom(options)` or `Component.joinTo(options)` was used to create an explicit directional join.

Condition Object

A condition narrows the query results. The `query.Condition` object performs the same function as the `search.Filter` object in the [N/search Module](#). The primary difference is that `query.Condition` objects can contain other `query.Condition` objects.

To create conditions:

- Use `Query.createCondition(options)` to create conditions for the initial query definition created with `query.create(options)`.
- Use `Component.createCondition(options)` to create conditions for the join relationships created with `Query.autoJoin(options)`, `Query.joinFrom(options)/Query.joinTo(options)`, `Component.autoJoin(options)`, or `Component.joinFrom(options)/Component.joinTo(options)`.
- If you have multiple conditions, use `Query.and()`, `Query.or()`, and `Query.not()` to create a new nested condition.
- If you want to use a formula to define your conditions, assign the formula to `Condition.formula`.

- Assign your simple or nested conditions as array values to [Query.condition](#).

RelativeDate Object

The [query.RelativeDate](#) object represents a date that is relative to the current date. You can use relative dates when you create query conditions.

To create relative dates:

- Use [query.createRelativeDate\(options\)](#) to create a [query.RelativeDate](#) object. When you call [query.createRelativeDate\(options\)](#), use the values in the [query.DateId](#) enum to specify a date that is relative to the current date.
- Use [Query.createCondition\(options\)](#) or [Component.createCondition\(options\)](#) to create a condition using the [query.RelativeDate](#) object. Alternatively, you can create a condition using values in the [query.RelativeDateRange](#) enum.
- If you have multiple conditions, use [Query.and\(\)](#), [Query.or\(\)](#), and [Query.not\(\)](#) to create a new nested condition.
- Assign your simple or nested conditions as array values to [Query.condition](#).

Column Object

The [query.Column](#) object is the equivalent of the [search.Column](#) object in the [N/search Module](#). The [query.Column](#) object describes the field types (columns) that are displayed from the query results.

To create columns:

- Use [Query.createColumn\(options\)](#) to create a column on the initial query definition created with [query.create\(options\)](#).
- Use [Component.createColumn\(options\)](#) to create a column on a join relationship created with [Query.autoJoin\(options\)](#), [Query.joinFrom\(options\)/Query.joinTo\(options\)](#), [Component.autoJoin\(options\)](#), or [Component.joinFrom\(options\)/Component.joinTo\(options\)](#).
- If you want to use a formula to define your columns, assign the formula to [Column.formula](#).
- Assign all created columns as array values to [Query.columns](#).

Sort Object

The [query.Sort](#) object describes how query results are sorted (for example, ascending or descending, case sensitive or case insensitive, and so on).

To create a sort:

- Use [Query.createSort\(options\)](#) to create a sort on the initial query definition created with [query.create\(options\)](#).
- Use [Component.createSort\(options\)](#) to create a sort based on a join relationship created with [Query.autoJoin\(options\)](#), [Query.joinFrom\(options\)/Query.joinTo\(options\)](#), [Component.autoJoin\(options\)](#), or [Component.joinFrom\(options\)/Component.joinTo\(options\)](#).
- Assign all created sorts as array values to [Query.sort](#).

ResultSet and Result Objects

When you are ready to execute your query, call [Query.run\(\)](#). This method returns a [query.ResultSet](#) object, which encapsulates the metadata for the set of results returned by the query.

To access your actual query results, iterate through the [ResultSet.results](#) array. Each member of the [ResultSet.results](#) array is a [query.Result](#) object. The [query.Result](#) object encapsulates a single row of the result set.

Page, PagedData, and PageRange Objects

You also can execute your query by calling [Query.runPaged\(\)](#). This method returns a [query.PagedData](#) object, which encapsulates a set of paged query results.

To access your query results, iterate through the paged query results using [PagedData.iterator\(\)](#). You can access each page of the query results, which are represented by [query.Page](#) objects. The [query.PageRange](#) object encapsulates the range of query results for a page.

N/query Module Terminology

Term	Definition	For More Information
Aggregate function	An aggregate function performs a calculation on a column of values and returns a single value. You can add aggregate functions to conditions and query results columns.	<ul style="list-style-type: none"> ■ query.Aggregate ■ Component.createColumn(options) ■ Component.createCondition(options) ■ Query.createColumn(options) ■ Query.createCondition(options)
Column	A column describes the field types (columns) that are displayed from the query results. A column is also known as a query results column.	<ul style="list-style-type: none"> ■ query.Column
Component	<p>When you script queries with the N/query module, your query is made up of one or more components, which are represented as query.Component objects. You can think of a component as a building block; each new component builds on the previous component created.</p> <ul style="list-style-type: none"> ■ The first component created represents the initial search type and is a child of query.Query. ■ Each subsequent component created is a child of the previous component. ■ The last component created encapsulates the join relationship between it and all of its parent components. <p>A query always contains at least one component: the root component. When you create the initial query definition using query.create(options), the root component is created automatically. Queries with joins contain multiple components. A new component is created each time you create a join using one of the following methods:</p> <ul style="list-style-type: none"> ■ Query.autoJoin(options), Query.joinFrom(options), or Query.joinTo(options) ■ Component.autoJoin(options), Component.joinFrom(options), or Component.joinTo(options) 	<ul style="list-style-type: none"> ■ query.Component
Condition	A condition narrows the query results.	<ul style="list-style-type: none"> ■ query.Condition
Formula	Formulas can be used to create conditions and columns.	<ul style="list-style-type: none"> ■ Formulas in Search ■ SQL Expressions
Group	You can summarize your query results into unique groups of column values.	<ul style="list-style-type: none"> ■ Column.groupBy
Join	A join lets you create a query based on a field type that is shared between two record types. You can use Query.autoJoin(options)	<ul style="list-style-type: none"> ■ query.Query ■ query.Component

Term	Definition	For More Information
	and <code>Component.autoJoin(options)</code> to create a join relationship automatically based on a field that you specify. You can use <code>Query.joinFrom(options)/Query.joinTo(options)</code> and <code>Component.joinFrom(options)/Component.joinTo(options)</code> to create explicit directional join relationships from one component to another.	
Page	A page represents one page from a set of paged query results. When you create a query with the N/query module, you can return the results as one result set or a set of paged results.	<ul style="list-style-type: none"> ■ <code>Query.runPaged()</code> ■ <code>query.PagedData</code> ■ <code>query.PageRange</code> ■ <code>query.Page</code>
Paged data	Paged data represents a set of paged query results.	<ul style="list-style-type: none"> ■ <code>Query.runPaged()</code> ■ <code>query.PagedData</code> ■ <code>query.PageRange</code> ■ <code>query.Page</code>
Page range	A page range is a set of pages from a set of paged query results.	<ul style="list-style-type: none"> ■ <code>Query.runPaged()</code> ■ <code>query.PagedData</code> ■ <code>query.PageRange</code> ■ <code>query.Page</code>
Relative date	A relative date is a date that is relative to the current date. You can use relative dates when you create query conditions.	<ul style="list-style-type: none"> ■ <code>query.RelativeDate</code> ■ <code>query.createRelativeDate(options)</code> ■ <code>query.RelativeDateRange</code>
Result	A result is a single row from a result set.	<ul style="list-style-type: none"> ■ <code>Query.run()</code> ■ <code>query.ResultSet</code> ■ <code>query.Result</code>
Result set	A result set is a set of query results.	<ul style="list-style-type: none"> ■ <code>Query.run()</code> ■ <code>query.ResultSet</code> ■ <code>query.Result</code>
Query definition	The query definition is the initial search type you define, plus any subsequent joins you define. The initial query definition is created with <code>query.create(options)</code> .	<ul style="list-style-type: none"> ■ <code>query.Query</code>
Query type	The query type is the initial query type of your query definition. It represents the record type you want to query for. It is set with the <code>query.Type</code> enum during the execution of <code>query.create(options)</code> . For example, if you want to query for customer records, specify <code>query.Type.CUSTOMER</code> as the query type when you call <code>query.create(options)</code> .	<ul style="list-style-type: none"> ■ <code>query.Query</code> ■ <code>query.Type</code>
Sort	A sort is placed on a query results column to describe how the query results are sorted (for example, ascending or descending, case sensitive or case insensitive, and so on).	<ul style="list-style-type: none"> ■ <code>query.Sort</code> ■ <code>Query.createSort(options)</code> ■ <code>Component.createSort(options)</code>

Formulas in the N/query Module

When you create a query using the N/query module, you can specify columns and conditions for the query. Columns describe the field types (or columns) that are displayed from the query results, and conditions narrow the query results based on certain criteria. You create a column using

`Query.createColumn(options)`, and you create a condition using `Query.createCondition(options)`. Both of these methods let you create a column or condition in two ways:

- Use the `fieldId` parameter to explicitly specify the field on which to create the column or condition.
- Use the `formula` parameter to specify a formula to create the column or condition.

You can use formulas to perform a calculation to determine the column or condition value based on the values of other fields in the record. For example, consider a situation in which you are working with Customer records that include custom fields. These custom fields contain the amount of stock for various items (50 units of item A, 24 units of item B, and so on). In your query results, you want to include a column that calculates and displays the total amount of stock for all items for a Customer. If the Customer records include three custom stock fields, you can create the result column as follows:

```
1 | query.createColumn({
2 |   formula: '{item_A_stock} + {item_B_stock} + {item_C_stock}',
3 |   type: query.ReturnType.INTEGER
4 | });
```

When you use a formula to create a column or condition, you must also use the `type` parameter to specify the return type of the formula. This parameter accepts values from the `query.ReturnType` enum. Defining the formula's return type might be required if the return type cannot be determined automatically based on the formula. When you set the `type` parameter, the return value is properly formatted based on the data type that you specify.

For more information on formulas, see the help topics [Formulas in Search](#) and [SQL Expressions](#).

Formulas in Joined Queries

You can join your queries with other record types. Joining queries lets you obtain and display query results with field values from multiple record types. When you use a formula in a joined query, you must use fully qualified field IDs to access the fields in each joined record type. You must specify the full join trail from the base record type. The join trail differs depending on the record types and join type.

Use the `^` and `<` operators to access fields in joined queries. You can use these operators when working with formulas in SuiteScript or the NetSuite UI. Use the `^` operator to access fields in record types that are joined using `Query.joinTo(options)` or `Component.joinTo(options)`. This type of join is also known as a polymorphic join. Use the `<` operator to access fields in record types that are joined using `Query.joinFrom(options)` or `Component.joinFrom(options)`. This type of join is also known as an inverse join. When you use `Query.autoJoin(options)` or `Component.autoJoin(options)`, you do not need to use the `^` or `<` operators to access fields in the joined query.

The following table lists common join operations and the corresponding join trail.

Join Type	Join Operation	Join Trail
Automatic using <code>Query.autoJoin(options)</code> or <code>Component.autoJoin(options)</code>	<pre>1 // The base record type is Customer 2 var myCustomerQuery = query.create({ 3 type: query.Type.CUSTOMER 4 }); 5 6 // The joined record type is Employee 7 var mySalesRepJoin = myCustomer 8 Query.autoJoin({ 9 fieldId: 'salesrep' 10});</pre>	<p>Base record fields (Customer)</p> <ul style="list-style-type: none"> ■ <code>customer.<baseFieldName></code> ■ Example: <code>customer.email</code> <p>Joined record fields (Employee)</p> <ul style="list-style-type: none"> ■ <code>customer.salesrep.<joinedFieldName></code> ■ Example: <code>customer.salesrep.phone</code>
Polymorphic using <code>Query.joinTo(options)</code> or <code>Component.joinTo(options)</code>	<pre>1 // The base record type is Transaction 2 var myTransactionQuery = query.create({ 3 type: query.Type.TRANSACTION 4});</pre>	<p>Base record fields (Transaction)</p> <ul style="list-style-type: none"> ■ <code>transaction.<baseFieldName></code>

Join Type	Join Operation	Join Trail
	<pre> 4 }); 5 6 // The joined record type is Employee 7 var myEmployeeJoin = myTransaction 8 Query.joinTo({ 9 fieldId: 'createdby', 10 target: 'employee' 11 }); </pre>	<ul style="list-style-type: none"> Example: transaction.entity <p>Joined record fields (Employee)</p> <ul style="list-style-type: none"> transaction.<baseFieldName>^employee.<joinedFieldName> Example: transaction.createdby^employee.email
Inverse using Query.joinFrom(options) or Component.joinFrom(options)	<pre> 1 // The base record type is Employee 2 var myEmployeeQuery = query.create({ 3 type: query.Type.EMPLOYEE 4 }); 5 6 // The joined record type is Transaction 7 var myTransactionJoin = myEmployee 8 Query.joinFrom({ 9 fieldId: 'entity', 10 source: 'transaction' 11 }); </pre>	<p>Base record fields (Employee)</p> <ul style="list-style-type: none"> employee.<baseFieldName> Example: employee.entityid <p>Joined record fields (Transaction)</p> <ul style="list-style-type: none"> employee.<baseFieldName><transaction.daysoverduesearch> Example: employee.entity<transaction.daysoverduesearch>

Join Trail Formatting

When you use join trails to access fields in joined queries, you can add whitespace characters and parentheses to improve the readability of your formulas. For example, consider this join trail:

employee.entity<transaction.daysoverduesearch>

The following join trails are equivalent to this one:

- employee.entity < transaction.daysoverduesearch>
- employee.(entity<transaction>).daysoverduesearch

Relative Dates in the N/query Module

You can use relative dates when you create query conditions. The [query.RelativeDate](#) object represents a specific date or moment in time relative to the current date. Each of the values in the [query.RelativeDateRange](#) enum represents a range of dates relative to the current date. When you use a [query.RelativeDate](#) object or [query.RelativeDateRange](#) enum value to create a query condition, make sure that you use an operator that makes sense for the relative date that you provide to [Query.createCondition\(options\)](#) or [Component.createCondition\(options\)](#). The [query.Operator](#) enum contains the supported operators for the N/query module, but not all operators apply to relative dates. Use the following operators with relative dates:

- AFTER
- AFTER_NOT
- BEFORE
- BEFORE_NOT
- ON
- ON_NOT
- ON_OR_AFTER
- ON_OR_AFTER_NOT
- ON_OR_BEFORE

- ON_OR_BEFORE_NOT
- WITHIN
- WITHIN_NOT

When you create a query condition using the WITHIN or WITHIN_NOT operators and a query.RelativeDate object, the condition uses the current date as one of the boundaries of the date range. For example, consider the following query.RelativeDate object that represents a date two days before the current date:

```

1 var myDatesAgo = query.createRelativeDate({
2   dateId: query.DateId.DAYS_AGO,
3   value: 2
4 });

```

You can use this myDatesAgo object when you create a query condition. Consider the following query condition that is created using the WITHIN operator and this myDatesAgo object:

```

1 var myCondition = myQuery.createCondition({
2   fieldId: 'trandate',
3   operator: query.Operator.WITHIN,
4   values: myDatesAgo
5 });

```

This query condition matches dates that are between two days ago and the current date (the day before yesterday, yesterday, and today).

Conversely, consider the following query.RelativeDate object that represents a date two days after the current date:

```

1 var myDatesFromNow = query.createRelativeDate({
2   dateId: query.DateId.DAYS_FROM_NOW,
3   value: 2
4 });

```

If you create a query condition using the WITHIN operator and this myDatesFromNow object, the condition matches dates that are between the current date and two days from now (today, tomorrow, and the day after tomorrow).

You can use the query.RelativeDate object, the [query.RelativeDateRange](#) enum, and the WITHIN operator to specify complex date ranges. You can do this in several ways:

- Use a single query.RelativeDate object or [query.RelativeDateRange](#) enum value. When you use a single query.RelativeDate object, the object represents a specific moment in time, so the current date is used automatically as one of the boundaries. When you use a single query.RelativeDateRange enum value, the enum value represents a range of dates, so the start and end properties of the date range are used automatically as the boundaries. For example:

```

1 var myComplexCondition = myQuery.createCondition({
2   fieldId: 'trandate',
3   operator: query.Operator.WITHIN,
4   values: query.RelativeDateRange.SAME_DAY_LAST_WEEK
5 });

```

In this example, the first boundary is the beginning of the same day last week, and the second boundary is the end of the same day last week. Using query.RelativeDateRange.SAME_DAY_LAST_WEEK is equivalent to using either of the following:

- query.RelativeDateRange.SAME_DAY_LAST_WEEK.interval

- [query.RelativeDateRange.SAME_DAY_LAST_WEEK.start, query.RelativeDateRange.SAME_DAY_LAST_WEEK.end]
- Use the start and end properties of values in the `query.RelativeDateRange` enum directly in the values parameter for `Query.createCondition(options)` or `Component.createCondition(options)`. For example:

```

1 | var myComplexCondition = myQuery.createCondition({
2 |   fieldId: 'trandate',
3 |   operator: query.Operator.WITHIN,
4 |   values: [query.RelativeDateRange.THIS_FISCAL_YEAR.start, query.RelativeDateRange.YESTERDAY.end]
5 | });

```

- Use a combination of `query.RelativeDateRange` enum values and custom `query.RelativeDate` objects. For example:

```

1 | var myEndDate = query.createRelativeDate({
2 |   dateId: query.DateId.WEEKS_AGO,
3 |   value: 2
4 | });
5 |
6 | var myComplexCondition = myQuery.createCondition({
7 |   fieldId: 'trandate',
8 |   operator: query.Operator.WITHIN,
9 |   values: [query.RelativeDateRange.THREE_FISCAL_YEARS_AGO.start, myEndDate]
10 | });

```

SuiteQL in the N/query Module

SuiteQL is a query language based on the SQL-92 revision of the SQL database query language. It provides advanced query capabilities you can use to access your NetSuite records and data. For more information about SuiteQL, including limitations, exceptions, and more usage examples, see the help topic [SuiteQL](#).

In SuiteScript, you can create and run SuiteQL queries using the N/query module. Queries created using SuiteQL can be more powerful and flexible than queries created using other APIs in the N/query module. SuiteQL queries can also provide the best query performance for many use cases. You can create your own SuiteQL query strings, which lets you design and run complex SQL queries that cannot be created otherwise. For examples of SuiteQL queries you can create, see [Examples of Using SuiteQL in the N/query Module](#).



Important: To create your own SuiteQL query strings, you must know the names of the record types and fields you want to use. For more information about finding record type and field names, see the help topic [Finding Record Type and Field Names](#).

Using the N/query module, you can run SuiteQL queries in the following ways:

- Convert an existing query (as a `query.Query` object) to its SuiteQL representation (as a `query.SuiteQL` object) and run the query. To learn more, see [Converting an Existing Query to SuiteQL](#).
- Run an arbitrary SuiteQL query as a paged or non-paged query. To learn more, see [Running an Arbitrary SuiteQL Query](#).

Converting an Existing Query to SuiteQL

If you already have a `query.Query` object in your script (one that you created using `query.create(options)` or loaded using `query.load(options)`), you can convert the query to its SuiteQL representation. The `Query.toSuiteQL()` method converts a `query.Query` object to a `query.SuiteQL` object. The resulting

[query.SuiteQL](#) object represents the same query as the original [query.Query](#) object and, when run, returns the same query results.



Important: The resulting SuiteQL query string (contained in the [SuiteQL.query](#) property) does not include any aliases you set on query result columns in the original [query.Query](#) object. For more information about aliases, see [Column.alias](#).

A [query.SuiteQL](#) object includes the following properties:

- [SuiteQL.columns](#) — The result columns to be returned from the query. This property is an array of [query.Column](#) objects. The value of this property is the same as the value of the [Query.columns](#) property in the original [query.Query](#) object.
- [SuiteQL.params](#) — The parameters for the query. In SuiteQL, query conditions are represented using the WHERE clause and a set of parameters.
- [SuiteQL.query](#) — The string representation of the SuiteQL query. This string can contain SQL clauses, record or table names, field names, operators, and more.
- [SuiteQL.type](#) — The type of the query. This property uses values from the [query.Type](#) enum. The value of this property is the same as the value of the [Query.type](#) property in the original [query.Query](#) object.

To run the SuiteQL query, use one of the following methods:

- Use [SuiteQL.run\(\)](#) to run the query as a non-paged query. This method returns the results as a [query.ResultSet](#) object.
- Use [SuiteQL.runPaged\(options\)](#) to run the query as a paged query. This method returns the results as a [query.PagedData](#) object. The default page size is 50 results per page. The minimum page size is 5 results per page, and the maximum page size is 1000 results per page.

The following example shows you how to create a query as a [query.Query](#) object, convert the query to SuiteQL, and run the resulting SuiteQL query as a non-paged query:

```

1 var myCustomerQuery = query.create({
2   type: query.Type.CUSTOMER
3 });
4
5 myCustomerQuery.columns = [
6   myCustomerQuery.createColumn({
7     fieldId: 'entityid'
8   }),
9   myCustomerQuery.createColumn({
10    fieldId: 'email'
11  })
12 ];
13
14 myCustomerQuery.condition = myCustomerQuery.createCondition({
15   fieldId: 'isperson',
16   operator: query.Operator.IS,
17   values: [true]
18 });
19
20 var mySQLCustomerQuery = myCustomerQuery.toSuiteQL();
21
22 var results = mySQLCustomerQuery.run();

```

In the preceding example, the `mySQLCustomerQuery` variable contains the resulting [query.SuiteQL](#) object after the conversion. In this object, the [SuiteQL.query](#) property contains the string representation of the original query. In this example, this property contains the following string:

```

1 SELECT customer.entityid AS entityidRAW /*{entityid#RAW}*/, customer.email AS emailRAW /*{email#RAW}*/ FROM customer WHERE
  customer.isperson = ?

```

The `SuiteQL.params` property contains the parameter value true. When the SuiteQL query runs, the question mark (?) in the query string is replaced with the parameter value (true).

Running an Arbitrary SuiteQL Query

You can design your own SuiteQL queries and run them. The `query.runSuiteQL(options)` method lets you run an arbitrary SuiteQL query, and it returns query results as a `query.ResultSet` object. The `query.runSuiteQLPaged(options)` method lets you run a SuiteQL query as a paged query, and it returns query results as a `query.PagedData` object.

To specify the SuiteQL query to run, you can provide one of the following to `query.runSuiteQL(options)` or `query.runSuiteQLPaged(options)`:

- A string representation of the SuiteQL query

```
1 | var results = query.runSuiteQL({
2 |   query: 'SELECT customer.entityid, customer.email FROM customer'
3 | });

```

- A `query.SuiteQL` object

```
1 | // In this example, mySuiteQLCustomerQuery is an existing SuiteQL object
2 | var results = query.runSuiteQL(mySuiteQLCustomerQuery);

```

- A JavaScript Object that contains a `query` property and, optionally, a `params` property:

```
1 | var results = query.runSuiteQL({
2 |   query: 'SELECT customer.entityid, customer.email FROM customer WHERE customer.isperson = ?',
3 |   params: [true]
4 | });

```

Examples of Using SuiteQL in the N/query Module

The following examples demonstrate how to create and run SuiteQL queries using the features in the N/query module.

Convert an Existing Query to SuiteQL

This example RESTlet loads a SuiteAnalytics Workbook query with an ID of `customworkbook237` and runs it. The RESTlet then converts the query to SuiteQL and runs it. The RESTlet returns both sets of query results.

```
1 /**
2  * @NApiVersion 2.x
3  * @NScriptType restlet
4 */
5 define(['N/query'], function(query) {
6   return {
7     get: function(context) {
8       // Load the workbook by name (record ID)
9       var openSalesOrders = query.load('customworkbook237');
10
11      // Run the query
12      var resultQuery = openSalesOrders.run();
13
14      // Convert the query to its SuiteQL representation
15      var openSalesOrdersSQL = openSalesOrders.toSuiteQL();
16
17      // Examine the SuiteQL query string

```

```

18     var suiteQL = openSalesOrdersQL.query;
19
20     // Run the SuiteQL query
21     var resultSuiteQL = query.runSuiteQL(suiteQL);
22
23     // Compose the RESTlet response
24     var response = {
25         query: openSalesOrders,
26         resultQuery: resultQuery,
27         suiteQL: suiteQL,
28         resultSuiteQL: resultSuiteQL
29     };
30
31     // Return the response
32     return JSON.stringify(response);
33 }
34 });
35 });

```

Create a Custom SuiteQL Query

This example RESTlet constructs a custom query string using SuiteQL, runs the query, and returns the query results.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType restlet
4 */
5 define(['N/query'], function(query) {
6     return {
7         get: function(context) {
8             // Construct the SuiteQL query string
9             var suiteQL =
10                "SELECT " +
11                "\"TRANSACTION\".tranid AS tranidRAW /*{tranid#RAW}*/, " +
12                "\"TRANSACTION\".trandate AS trandateRAW /*{trandate#RAW}*/, " +
13                "\"TRANSACTION\".postingperiod AS postingperiodDISPLAY /*{postingperiod#DISPLAY}*/, " +
14                "BUILTIN.DF(\"TRANSACTION\".postingperiod) AS postingperiodDISPLAY /*{postingperiod#DISPLAY}*/, " +
15                "BUILTIN.DF(\"TRANSACTION\".status) AS statusDISPLAY /*{status#DISPLAY}*/, " +
16                "BUILTIN.DF(transactionLine.item) AS transactionlinesitemDISPLAY /*{transactionlines.item#DISPLAY}*/, " +
17                "BUILTIN.DF(\"TRANSACTION\".entity) AS entityDISPLAY /*{entity#DISPLAY}*/, " +
18                "transactionLine.quantity * -1 AS transactionlinesquantity /*- {transactionlines.quantity}*/, " +
19                "BUILTIN.CONOLIDATE(transactionLine.netamount, 'LEDGER', 'DEFAULT', 'DEFAULT', 1, 396, 'DEFAULT') AS transactionlinesnetamountCU /*{transactionlines.netamount#CURRENCY_CONOLIDATED}*/, " +
20                "BUILTIN.CURRENCY(BUILTIN.CONOLIDATE(transactionLine.netamount, 'LEDGER', 'DEFAULT', 'DEFAULT', 1, 396, 'DEFAULT')) AS transactionlinesnetamountCU_C /*{transactionlines.netamount#CURRENCY_CONOLIDATED}*/, " +
21                "CUSTOMRECORD41.custrecord14 AS custrecord15customrecord41c /*{custrecord15<customrecord41.custrecord14#RAW}*/ " +
22                "+ " +
23                "\"TRANSACTION\", " +
24                "+ " +
25                "CUSTOMRECORD41, " +
26                "+ " +
27                "\"ACCOUNT\"", " +
28                "+ " +
29                "TransactionAccountingLine, " +
30                "+ " +
31                "transactionLine " +
32                "+ " +
33                "WHERE " +
34                "+ " +
35                "(((" +
36                "TRANSACTION\".\"ID\" = CUSTOMRECORD41.custrecord15 AND TransactionAccountingLine.\"ACCOUNT\" = \"ACCOUNT\".\"ID" +
37                "(+)) AND (transactionLine.\"TRANSACTION\" = TransactionAccountingLine.\"TRANSACTION\" AND transactionLine.\"ID\" = TransactionAccountingLine.transactionline) AND \"TRANSACTION\".\"ID\" = transactionLine.\"TRANSACTION\")" +
38                "+ " +
39                "AND ((UPPER(\"TRANSACTION\".\"TYPE\") IN ('SALESORD') AND UPPER(\"TRANSACTION\".status) IN ('SALESORD:D', 'SALESORD:E', 'SALESORD:B') AND UPPER(\"ACCOUNT\".accttype) IN ('INCOME') AND (NOT (" +
40                "transactionLine.itemtype IN ('ShipItem') " +
41                "+ " +
42                "OR transactionLine.itemtype IS NULL) AND ((transactionLine.quantity * -1) - NVL(transactionLine.quantitycommitted, 0)) - NVL(transactionLine.quantityshiprev, 0) > 0" +
43                "AND NVL(transactionLine.mainline, 'F') = 'F' AND NVL(transactionLine.taxline, 'F') = 'F' AND NVL(transactionLine.isclosed, 'F') = 'F'))" +
44                "+ " +
45                "ORDER BY " +
46                "+ " +
47                "\"TRANSACTION\".trandate ASC NULLS LAST";
48
49     // Run the SuiteQL query
50     var resultSuiteQL = query.runSuiteQL(suiteQL);
51
52     // Compose the RESTlet response
53     var response = {
54         resultSuiteQL: resultSuiteQL
55     };
56
57     // Return the response
58     return JSON.stringify(response);
59 }

```

```
36 |     }
37 | });

```

Accept a SuiteQL Query as an Argument

This example RESTlet accepts a SuiteQL query as a provided argument to the get endpoint, runs the query, and returns the query results.

```
1 /**
2  * @NApiVersion 2.x
3  * @NScriptType restlet
4  */
5 define(['N/query'], function(query) {
6     return {
7         get: function(context) {
8             return runQuery(query, context);
9         },
10        post: function(context) {
11            return runQuery(query, context);
12        }
13    };
14
15    function runQuery(query, context) {
16        // Get the SuiteQL query string from the arguments. For example, the
17        // SuiteQL query may look like the following:
18        //
19        // &suiteQL=select%20type%2C%20BUILTIN.DF(type)%2C%20tranid%2C%20trandate%20from%20transaction%20where%20type%3D'SalesOrd'
20        var id = context.id;
21        var suiteQL = context.suiteQL;
22
23        // Run the query
24        var suiteQLResults = query.runSuiteQL(suiteQL);
25
26        // Compose the RESTlet response
27        var response = {
28            id: id,
29            suiteQL: suiteQL,
30            resultSuiteQL: suiteQLResults.asMappedResults()
31        };
32
33        // Return the response
34        return JSON.stringify(response);
35    };
36 });

```

Run a Paged SuiteQL Query

This example script runs a SuiteQL query as a paged query with a page size of 10 results per page.

```
1 /**
2  * @NApiVersion 2.x
3  */
4 require(['N/query'], function(query) {
5     // Construct the SuiteQL query string
6     var sql =
7         "SELECT " +
8         " scriptDeployment.primarykey, scriptexecutioncontextmap.executioncontext " +
9         " FROM " +
10        " scriptDeployment, scriptexecutioncontextmap " +
11        " WHERE " +
12        " scriptexecutioncontextmap.scriptrecord = scriptDeployment.primarykey " +
13        " AND " +
14        " scriptexecutioncontextmap.executioncontext IN ('WEBSTORE', 'WEBAPPLICATION')";
15
16     // Run the SuiteQL query as a paged query and return an iterator
17     var resultIterator = query.runSuiteQLPaged({

```

```

18     query: sql,
19     pageSize: 10
20   }).iterator();
21
22   // Use the iterator to process each page of results
23   resultIterator.each(function(page) {
24     var pageIterator = page.value.data.iterator();
25     pageIterator.each(function(row) {
26       log.debug('ID: ' + row.value.getValue(0) + ', Context: ' + row.value.getValue(1));
27       return true;
28     });
29
30     return true;
31   });
32 });

```

Use a SuiteQL Query in a Map/Reduce Script

This example map/reduce script uses a SuiteQL query as the source of input data. The value 271 is the internal ID of a transaction record.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType mapreduce
4 */
5 define(['N/query'], function(query) {
6   // Define the getInputData() stage function
7   function getInputData() {
8     // Construct the SuiteQL query string
9     var suiteQL =
10       "SELECT " +
11       " \\"TRANSACTION\\".tranid AS tranidRAW /*{tranid#RAW}*/, " +
12       " \\"TRANSACTION\\".trandate AS trandateRAW /*{trandate#RAW}*/, " +
13       " \\"TRANSACTION\\".postingperiod AS postingperiodDISPLAY /*{postingperiod#DISPLAY}*/ " +
14       "FROM " +
15       " \\"TRANSACTION\\" WHERE \\"TRANSACTION\\\".\\\"ID\\\" = ? ";
16
17   // Return the query results as input data. The value 271 is the
18   // internal ID of a transaction record
19   return {
20     type: 'suiteql',
21     query: suiteQL,
22     params: [271]
23   };
24 }
25
26 // Define the map() stage function
27 function map(context) {
28   context.write(context.key, context.value);
29 }
30
31 // Define the reduce() stage function
32 function reduce(context) {
33   context.write(context.key, context.values[0]);
34 }
35
36 // Define the summarize() stage function
37 function summarize(summary) {
38   if (summary.inputSummary.error) {
39     log.debug('An error occurred.');
40   }
41 }
42
43 // Return the function definitions
44 return {
45   getInputData: getInputData,
46   map: map,
47   reduce: reduce,
48   summarize: summarize
49 }

```

50 |});

query.Column



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>A query result column.</p> <p>The <code>query.Column</code> object is the equivalent of the <code>search.Column</code> object in the N/search Module. The <code>query.Column</code> object describes the field types (columns) that are displayed from the query results.</p> <p>To create columns:</p> <ul style="list-style-type: none"> ■ Use Query.createColumn(options) to create a column on the initial query definition created with query.create(options). ■ Use Component.createColumn(options) to create a column on a join relationship created with Query.autoJoin(options) or Component.autoJoin(options). ■ Assign all created columns as array values to Query.columns. For an example, see Syntax.
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/query Module
Methods and Properties	Column Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   myCustomerQuery.createColumn({
16     fieldId: 'id'
17   }),
18   mySalesRepJoin.createColumn({
19     fieldId: 'entityid'
20   }),
21   mySalesRepJoin.createColumn({
22     fieldId: 'email'
23   }),
24   mySalesRepJoin.createColumn({

```

```

25     fieldId: 'hiredate'
26   )),
27 ];
28
29 myCustomerQuery.sort = [
30   myCustomerQuery.createSort({
31     column: myCustomerQuery.columns[1]
32   )),
33   mySalesRepJoin.createSort({
34     column: mySalesRepJoin.columns[0],
35     ascending: false
36   ))
37 ];
38
39 var resultSet = myCustomerQuery.run();
40 ...
41 // Add additional code

```

Column.aggregate



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An aggregate function that is performed on the query result column. An aggregate function performs a calculation on the column values and returns a single value. This property is set when Query.createColumn(options) or Component.createColumn(options) is executed. For a list of supported aggregate functions, see the query.Aggregate enum.
Type	string (read-only)
Module	N/query Module
Parent Object	query.Column
Sibling Object Members	Column Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myAggColumn = myTransactionQuery.createColumn({
8   fieldId: 'amount',
9   aggregate: query.Aggregate.AVERAGE
10 });
11
12 myTransactionQuery.columns = [myAggColumn];
13
14 var theAggregate = myAggColumn.aggregate;
15 ...
16 // Add additional code

```

Column.alias



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An alias for this column. An alias is an alternate name for a column, and the alias is used in mapped results. In general, the alias is an optional property. If you want to use mapped results in your script, the alias is required. To use mapped results, you must specify an alias in the following situations: <ul style="list-style-type: none">■ You must specify an alias for a column when the column uses a formula.■ You must specify an alias when two columns in a joined query use the same field ID. For example, many record types include the entity field ID. If you join two record types that use the entity field ID, and you use the entity field ID to create result columns for both record types, you must specify an alias for one of the columns. This alias distinguishes the two columns that have the same field ID. This property is set when Query.createColumn(options) or Component.createColumn(options) is executed.
Type	string
Module	N/query Module
Parent Object	query.Column
Sibling Object Members	Column Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myAliasColumn = myTransactionQuery.createColumn({
8   fieldId: 'amount',
9   aggregate: query.Aggregate.AVERAGE,
10  alias: 'sunkcostamount'
11 });
12
13 myTransactionQuery.columns = [myAliasColumn];
14
15 var theAlias = myAliasColumn.alias;
16 ...
17 // Add additional code

```

Column.component



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A reference to the query.Component object to which this query result column belongs.
-----------------------------	--

	This property is set when Query.createColumn(options) or Component.createColumn(options) is executed.
Type	query.Component (read-only)
Module	N/query Module
Parent Object	query.Column
Sibling Object Members	Column Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myAmountColumn = myTransactionQuery.createColumn({
8   fieldId: 'amount'
9 });
10
11 var theComponent = myAmountColumn.component;
12 ...
13 // Add additional code

```

Column.context



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The field context for values in the query result column.</p> <p>This property is set when Query.createColumn(options) or Component.createColumn(options) is executed. The field context determines how field values are displayed in a column. For example, you can specify that a column should display raw data (such as internal IDs), consolidated or converted amounts (such as currency totals), or user-friendly values (such as names).</p> <p>This property is an Object that includes the name of the context (which is a value in the query.FieldContext enum) and any parameters that apply to that context. In this release, only the <code>query.FieldContext.CONVERTED</code> context uses parameters. For information about these parameters, see Query.createColumn(options) or Component.createColumn(options).</p>
Type	Object (read-only)
Module	N/query Module
Parent Object	query.Column
Sibling Object Members	Column Object Members
Since	2019.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myTranLinesJoin = myTransactionQuery.autoJoin({
8   fieldId: 'transactionlines'
9 });
10
11 myTransactionQuery.condition = myTranLinesJoin.createCondition({
12   fieldId: 'netamount',
13   operator: query.Operator.GREATER,
14   values: 50000
15 });
16
17 myTransactionQuery.columns = [
18   myTranLinesJoin.createColumn({
19     fieldId: 'netamount'
20   })
21 ];
22
23 var unconsolidatedResultSet = myTransactionQuery.run();
24
25 // Log unconsolidated amounts
26 for (var i in unconsolidatedResultSet.results)
27   log.debug(unconsolidatedResultSet.results[i].values[0]);
28
29 myTransactionQuery.columns = [
30   myTranLinesJoin.createColumn({
31     fieldId: 'netamount',
32     context: query.FieldContext.CURRENCY_CONSOLIDATED
33   })
34 ];
35
36 var consolidatedResultSet = myTransactionQuery.run();
37
38 // Log consolidated amounts
39 for (var i in consolidatedResultSet.results)
40   log.debug(consolidatedResultSet.results[i].values[0]);
41 ...
42 // Add additional code

```

Column.fieldId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of the query result column. This property is set during the execution of Query.createColumn(options) or Component.createColumn(options) . This property and the Column.formula property cannot be set at the same time.
Type	string (read-only)
Module	N/query Module
Parent Object	query.Column

Sibling Object Members	Column Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myAmountColumn = myTransactionQuery.createColumn({
8   fieldId: 'amount'
9 });
10
11 var theFieldId = myAmountColumn.fieldId;
12 ...
13 // Add additional code

```

Column.formula

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A formula used to create the query result column. This property is set during the execution of Query.createColumn(options) or Component.createColumn(options) . This property and the Column.fieldId property cannot be set at the same time. For more information on formulas, see the help topics Formulas in Search and SQL Expressions .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Column
Sibling Object Members	Column Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6

```

```

7  var myFormulaColumn = myTransactionQuery.createColumn({
8    type: query.ReturnType.CURRENCY,
9    formula: '{amount} * 125'
10   });
11
12  var theFormula = myFormulaColumn.formula;
13 ...
14 // Add additional code

```

Column.groupBy

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether the query results are grouped by this query result column. This property is set during the execution of Query.createColumn(options) or Component.createColumn(options) .
Type	boolean (read-only)
Module	N/query Module
Parent Object	query.Column
Sibling Object Members	Column Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var myGroupByColumn = myCustomerQuery.createColumn({
8   fieldId: 'currency',
9   groupBy: true
10  });
11
12 var theGroupBy = myGroupByColumn.groupBy;
13 ...
14 // Add additional code

```

Column.label

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The label for the column. A label is important if the query object is used as the data source for printing (for example, in the TemplateRenderer.addQuery(options) method in the N/render module).
Type	string (read-only)
Module	N/query Module

Parent Object	query.Column
Sibling Object Members	Column Object Members
Since	2019.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var myLabelColumn = myCustomerQuery.createColumn({
8   fieldId: 'currency',
9   label: 'Dollars'
10 });
11
12 var theLabel = myLabelColumn.label;
13 ...
14 // Add additional code

```

Column.type

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The return type of the formula used to create the query result column. This property is set during the execution of Query.createColumn(options) or Component.createColumn(options) . If a formula is specified when these methods are called, this property contains the return type of the formula. If a formula is not specified, this property is null. For more information on formulas, see the help topics Formulas in Search and SQL Expressions .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Column
Sibling Object Members	Column Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...

```

```

3 | var myTransactionQuery = query.create({
4 |   type: query.Type.TRANSACTION
5 | });
6 |
7 | var myFormulaColumn = myTransactionQuery.createColumn({
8 |   type: query.ReturnType.CURRENCY,
9 |   formula: '{amount} * 125'
10| });
11|
12| var theFormulaType = myFormulaColumn.type;
13| ...
14| // Add additional code

```

query.Component

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>One component of the query definition. Each new component is created as a child to the previous component. All components exist as children to the query definition (query.Query).</p>
	<p>You can think of a component as a building block; each new component builds on the previous component created. The last component created encapsulates the relationship between it and all of its parent components.</p>
	<p>The query definition always contains at least one component. Queries with joins contain multiple components. The query definition (query.Query) contains a child <code>query.Component</code> object for each of the following:</p>
	<ul style="list-style-type: none"> ■ The initial query definition: The initial <code>query.Component</code> object is called the root component. It encapsulates the initial query type passed to query.create(options). The root component is automatically created with the query.Query object and is a child of the query.Query object. The Query.root property contains a reference to the root component. ■ The first join: The second <code>query.Component</code> object is created with Query.autoJoin(options). It encapsulates the relationship between the initial query definition and the second query type. This relationship is determined by the join ID passed to Query.autoJoin(options). The second <code>query.Component</code> object is a child of the root component. ■ Each subsequent join: The third <code>query.Component</code> object is created with Component.autoJoin(options). All subsequent joins and their respective <code>query.Component</code> objects are also created with Component.autoJoin(options). Each of these <code>query.Component</code> objects encapsulates the relationship between all previous query types and the new query type. This relationship is determined by the join ID passed to Component.autoJoin(options).
Supported Script Types	<p>Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/query Module
Methods and Properties	Component Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 | // Add additional code
2 | ...

```

```

3  var myCustomerQuery = query.create({
4    type: query.Type.CUSTOMER
5  });
6
7  var mySalesRepJoin = myCustomerQuery.autoJoin({
8    fieldId: 'salesrep'
9  });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   myCustomerQuery.createColumn({
16     fieldId: 'id'
17   }),
18   mySalesRepJoin.createColumn({
19     fieldId: 'entityid'
20   }),
21   mySalesRepJoin.createColumn({
22     fieldId: 'email'
23   }),
24   mySalesRepJoin.createColumn({
25     fieldId: 'hiredate'
26   }),
27 ];
28
29 myCustomerQuery.sort = [
30   myCustomerQuery.createSort({
31     column: myCustomerQuery.columns[1]
32   }),
33   mySalesRepJoin.createSort({
34     column: mySalesRepJoin.columns[0],
35     ascending: false
36   })
37 ];
38
39 var resultSet = myCustomerQuery.run();
40 ...
41 // Add additional code

```

Component.autoJoin(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a join relationship.</p> <p>Use the method query.create(options) to create your initial query definition (query.Query). The initial query definition uses one query type. For available query types, see query.Type.</p> <p>After you create the initial query definition, use Query.autoJoin(options) to create your first join (query.Component). Then use Component.autoJoin(options) to create each subsequent join (query.Component).</p> <div data-bbox="489 1531 1379 1647" style="border: 1px solid #f0e68c; padding: 10px;">  Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types. </div>
Returns	query.Component
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module

Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fieldId	string	required	<p>The ID of the field that joins the parent component to the new component.</p> <p>Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser.</p>

Errors

Error Code	Thrown If
RELATIONSHIP_ALREADY_USED	The specified join relationship already exists.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEntityJoin = myTransactionQuery.autoJoin({
8   fieldId: 'entity'
9 });
10
11 myTransactionQuery.columns = [
12   myEntityJoin.createColumn({
13     fieldId: 'subsidiary'
14   })
15 ];
16
17 myTransactionQuery.sort = [
18   myTransactionQuery.createSort({
19     column: myTransactionQuery.columns[0],
20     ascending: false
21   })
22 ];
23
24 var results = myTransactionQuery.runPaged({
25   pageSize: 10
26 });
27 ...
28 // Add additional code

```

Component.createColumn(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a query result column based on the query.Component object. The query.Column object is the equivalent of the search.Column object in the N/search Module. The query.Column object describes the field types (columns) that are displayed from the query results.</p>
	<p>To create columns:</p> <ul style="list-style-type: none"> ■ Use <code>Component.createColumn(options)</code> to create columns on the join relationships created with Query.autoJoin(options) and Component.autoJoin(options). Use this method in one of two ways: <ul style="list-style-type: none"> □ Pass in an argument for the parameter <code>options.fieldId</code>. □ Pass in an argument for the parameter <code>options.formula</code>. If you use this option, you can also use the optional parameter <code>options.type</code>. ■ If needed, use Query.createColumn(options) to create columns on the initial query definition created with query.create(options). ■ Assign all created columns as array values to Query.columns. For an example, see Syntax. <p>When you create a column, you can specify a field context. The field context determines how field values are displayed in the column. For example, you can specify that a column should display raw data (such as internal IDs), consolidated or converted amounts (such as currency totals), or user-friendly values (such as names). You can specify a field context in two ways:</p> <ul style="list-style-type: none"> ■ Use a context from the query.FieldContext enum directly as the value of the <code>options.context</code> parameter. For example: <pre> 1 myTransactionLine.createColumn({ 2 fieldId: 'netamount', 3 context: query.FieldContext.CURRENCY_CONOLIDATED 4 }); </pre> <p>This example is the simplest way to specify a field context that does not accept additional parameters. Because the <code>options.context</code> parameter is an Object, this example is equivalent to the following:</p> <pre> 1 myTransactionLine.createColumn({ 2 fieldId: 'netamount', 3 context: { 4 name: query.FieldContext.CURRENCY_CONOLIDATED 5 } 6 }); </pre> <ul style="list-style-type: none"> ■ Use a context from the query.FieldContext enum as the value of the <code>options.context.name</code> parameter, and specify additional parameters using the <code>options.context.params</code> parameter. For example: <pre> 1 myTransactionLine.createColumn({ 2 fieldId: 'netamount', 3 context: { 4 name: query.FieldContext.CONVERTED, 5 params: { 6 currencyId: 4, 7 date: new Date('2019/01/01') 8 } 9 } 10 }); </pre> <p>In this release, only the <code>query.FieldContext.CONVERTED</code> context uses additional parameters. The supported parameters are <code>currencyId</code> and <code>date</code>. For the <code>date</code> parameter, you can pass a JavaScript Date object or query.RelativeDate object.</p>

Returns	query.Column
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fieldId	string	required if options.formula is not used	The field ID of the query result column. This value sets the Column.fieldId property. Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser .
options.formula	string	required if options.fieldId is not used	The formula used to create the query result column. This value sets the Column.formula property. For more information on formulas, see the help topics Formulas in Search and SQL Expressions .
options.type	string	required if options.formula is used	If you use the options.formula parameter, use this parameter to explicitly define the formula's return type. This value sets the Column.type property. Use the appropriate query.ReturnType enum value to pass in your argument. This enum holds all the supported values for this parameter.
options.aggregate	string	optional	Use this parameter to run an aggregate function on your query result column. An aggregate function performs a calculation on the column values and returns a single value. This value sets the Column.aggregate property. Use the appropriate query.Aggregate enum value to pass in your argument. This enum holds all the supported values for this parameter.
options.groupBy	boolean	optional	Whether the query results are grouped by this query result column. This value sets the Column.groupBy property. If you do not pass in an argument, the default value is set to false.
options.label	string	optional	The label for the column.

Parameter	Type	Required / Optional	Description
			A label is important if the query object is used as the data source for printing (for example, in the TemplateRenderer.addQuery(options) method in the N/render module).
options.context	Object	optional	The field context for values in the query result column. This value sets the Column.context property.
options.context.name	string	required if options.context is used	The name of the field context. Use the appropriate query.FieldContext enum value to pass in your argument. This enum holds all the supported values for this parameter.
options.context.params	Object	required if options.context.name has a value of query.FieldContext.CONVERTED	The additional parameters to use with the specified field context. In this release, only the query.FieldContext.CONVERTED context uses additional parameters. The supported parameters are currencyId and date.
options.context.params.currencyId	number	required if options.context.name has a value of query.FieldContext.CONVERTED	The internal ID of the currency to convert to. You can specify the internal ID of any currency that is configured in your NetSuite account. For more information, see the help topic Multiple Currencies .
options.context.params.date	query.RelativeDate JavaScript Date	required if options.context.name has a value of query.FieldContext.CONVERTED	The date to use for the actual exchange rate between the base currency and the currency to convert to. For example, if you want to use the exchange rate that was in effect on March 3, 2019, specify a query.RelativeDate object or JavaScript Date object that represents this date.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.join({
8   fieldId: 'salesrep'
9 });
10
11
12 myCustomerQuery.columns = [
13   myCustomerQuery.createColumn({
14     fieldId: 'entityid'
15   }),
16   myCustomerQuery.createColumn({
17     fieldId: 'id'
18   }),
19   mySalesRepJoin.createColumn({
20     fieldId: 'entityid'
21   }),
22   mySalesRepJoin.createColumn({
23     fieldId: 'email'

```

```

24     }),
25     mySalesRepJoin.createColumn({
26       fieldId: 'hiredate'
27     }),
28   ];
29
30 myCustomerQuery.sort = [
31   myCustomerQuery.createSort({
32     column: myCustomerQuery.columns[1]
33   }),
34   mySalesRepJoin.createSort({
35     column: mySalesRepJoin.columns[0],
36     ascending: false
37   })
38 ];
39
40 var resultSet = myCustomerQuery.run();
41 ...
42 // Add additional code

```

Component.createCondition(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a condition (query filter) based on the query.Component object. A condition narrows the query results. The query.Condition object acts in the same capacity as the search.Filter object in the N/search Module. The primary difference is that query.Condition objects can contain other query.Condition objects.</p> <p>To create conditions:</p> <ul style="list-style-type: none"> ■ Use Component.createCondition(options) to create conditions on the join relationships created with Query.autoJoin(options) and Component.autoJoin(options). Use this method in one of two ways: <ul style="list-style-type: none"> □ Pass in arguments for the parameters <code>options.fieldId</code>, <code>options.operator</code>, and <code>options.values</code>. The combination of these arguments translates to <code><filter column><operator><field value></code> (for example, 'city' equals 'Boston'). □ Pass in an argument for the parameter <code>options.formula</code>. If you use this option, you can also use the optional parameter <code>options.type</code>. ■ If needed, use Query.createCondition(options) to create conditions on the initial query definition created with query.create(options). ■ If you have multiple conditions, use them to create a new nested condition with the methods Query.and(), Query.or(), and Query.not(). ■ Assign your simple or nested condition to Query.condition. For an example, see Syntax.
Returns	query.Condition
Supported Script Types	<p>Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/query Module
Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fieldId	string	required if options.operator and options.values are used	The field that the condition applies to. This value sets the Condition.fieldId property. Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser .
options.operator	string	required	The operator used by the condition. This value sets the Condition.operator parameter. Use the appropriate query.Operator enum value to pass in your argument. This enum holds all the supported values for this parameter.
options.values	string[] number[] boolean[] Date[] query.RelativeDate[] query.Period[]	required if options.fieldId and options.operator are used, and options.operator does not have a value of query.Operator.EMPTY or query.Operator.EMPTY_NOT	An array of values to use for the condition. This value sets the Condition.values property.
options.formula	string	required if options.fieldId is not used	The formula used to create the condition. This value sets the Condition.formula property. For more information on formulas, see the help topics Formulas in Search and SQL Expressions .
options.type	string	required if options.formula is used	If you use the options.formula parameter, use this parameter to explicitly define the formula's return type. This value sets the Condition.type property. Use the appropriate query.ReturnType enum value to pass in your argument. This enum holds all the supported values for this parameter.
options.aggregate	string	optional	An aggregate function to run on the condition. An aggregate function performs a calculation on the condition values and returns a single

Parameter	Type	Required / Optional	Description
			<p>value. This value sets the Condition.aggregate property.</p> <p>Use the appropriate query.Aggregate enum value to pass in your argument. This enum holds all the supported values for this parameter.</p>

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myLocationJoin = mySalesRepJoin.autoJoin({
12   fieldId: 'location'
13 });
14
15 var firstCondition = myCustomerQuery.createCondition({
16   fieldId: 'id',
17   operator: query.Operator.EQUAL,
18   values: 107
19 });
20 var secondCondition = myCustomerQuery.createCondition({
21   fieldId: 'id',
22   operator: query.Operator.EQUAL,
23   values: 2647
24 });
25 var thirdCondition = mySalesRepJoin.createCondition({
26   fieldId: 'email',
27   operator: query.Operator.START_WITH_NOT,
28   values: 'foo'
29 });
30
31 myCustomerQuery.condition = myCustomerQuery.and(
32   thirdCondition, myCustomerQuery.not(
33     myCustomerQuery.or(firstCondition, secondCondition)
34   )
35 );
36
37 var resultSet = search.run();
38 ...
39 // Add additional code

```

Component.createSort(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a sort based on the query.Component object. The query.Sort object describes a sort that is placed on a particular query result column or condition.
---------------------------	---

	<p>To create a sort:</p> <ul style="list-style-type: none"> ■ Use <code>Component.createSort(options)</code> to create a sort based on a join relationship created with <code>Query.autoJoin(options)</code> or <code>Component.autoJoin(options)</code>. ■ Use <code>Query.createSort(options)</code> to create a sort based on the initial query definition created with <code>query.create(options)</code>. ■ Assign all created sorts as array values to <code>Query.sort</code>. For an example, see Syntax.
Returns	<code>query.Sort</code>
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	<code>query.Component</code>
Sibling Object Members	<code>Component Object Members</code>
Since	2018.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.column</code>	<code>query.Column</code>	required	The query result column that you want to sort by. This value sets the <code>Sort.column</code> property.
<code>optionsascending</code>	boolean	optional	<p>Whether the sort direction is ascending. This value sets the <code>Sort.ascending</code> property.</p> <p>The default value of this property is <code>true</code>, meaning that the sort direction is ascending. If you want the sort direction to be descending, set this property to <code>false</code>.</p>
<code>options.caseSensitive</code>	boolean	optional	<p>Whether the sort is case sensitive. This value sets the <code>Sort.caseSensitive</code> property.</p> <p>If a sort is case sensitive (and the sort direction is ascending), rows with column values that start with uppercase letters are listed before rows with column values that start with lowercase letters. If a sort is not case sensitive, uppercase and lowercase letters are treated the same. For example, the following list of items is sorted using a case-sensitive sort with a sort direction of ascending:</p> <ul style="list-style-type: none"> ■ Banana ■ Orange ■ apple ■ grapefruit ■ kiwi

Parameter	Type	Required / Optional	Description
			<p>Here is the same list of items sorted using a regular (not case-sensitive) sort with a sort direction of ascending:</p> <ul style="list-style-type: none"> ■ apple ■ Banana ■ grapefruit ■ kiwi ■ Orange <p>The default value of this property is false.</p>
options.locale	string	optional	<p>The locale to use for the sort. This value sets the Sort.locale property.</p> <p>A locale represents a combination of language and region, and it can affect how certain values (such as strings) are sorted. For example, languages that share the same alphabet may sort characters differently. Use this property to ensure that query results are sorted using locale-specific rules.</p> <p>Use the appropriate query.SortLocale enum value to pass in your argument. This enum holds all the supported values for this parameter.</p>
options.nullsLast	boolean	optional	<p>Whether query results with null values are listed at the end of the query results. This value sets the Sort.nullsLast property.</p> <p>The default value of this property is the value of the options.ascending property. For example, if the options.ascending property is set to true, the options.nullsLast property is also set to true.</p>

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   myCustomerQuery.createColumn({
16     fieldId: 'id'
17   }),
18   mySalesRepJoin.createColumn({
19     fieldId: 'entityid'
20   }),

```

```

21 |     mySalesRepJoin.createColumn({
22 |         fieldId: 'email'
23 |     }),
24 |     mySalesRepJoin.createColumn({
25 |         fieldId: 'hiredate'
26 |     })
27 | );
28 |
29 | myCustomerQuery.sort = [
30 |     myCustomerQuery.createSort({
31 |         column: myCustomerQuery.columns[1]
32 |     }),
33 |     mySalesRepJoin.createSort({
34 |         column: mySalesRepJoin.columns[0],
35 |         ascending: false
36 |     })
37 | ];
38 |
39 | var resultSet = myCustomerQuery.run();
40 | ...
41 | // Add additional code

```

Component.join(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a join relationship. This method is an alias to Component.autoJoin(options).</p> <p>Use the method query.create(options) to create your initial query definition (query.Query). The initial query definition uses one query type. For available query types, see query.Type.</p> <p>After you create the initial query definition, use Query.autoJoin(options) to create your first join (query.Component). Then use Component.join(options) to create each subsequent join (query.Component).</p> <div data-bbox="502 1178 1380 1284" style="background-color: #ffffcc; border: 1px solid #ffcc00; padding: 10px;"> <p>Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types.</p> </div>
Returns	query.Component
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/query Module
Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.1

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description
options.fieldId	string	required	<p>The ID of the field that joins the parent component to the new component. This value determines the columns on which the components are joined and the type of the newly joined component.</p> <p>Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser.</p>

Errors

Error Code	Thrown If
RELATIONSHIP_ALREADY_USED	The specified join relationship already exists.

Syntax

 Important:	The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see N/query Module Script Samples .
---	--

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEntityJoin = myTransactionQuery.join({
8   fieldId: 'entity'
9 });
10
11 myTransactionQuery.columns = [
12   myEntityJoin.createColumn({
13     fieldId: 'subsidiary'
14   })
15 ];
16
17 myTransactionQuery.sort = [
18   myTransactionQuery.createSort({
19     column: myTransactionQuery.columns[0],
20     ascending: false
21   })
22 ];
23
24 var results = myTransactionQuery.runPaged({
25   pageSize: 10
26 });
27 ...
28 // Add additional code

```

Component.joinFrom(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates an explicit directional join relationship from another component to this component (an inverse join). This method sets the Component.source property on the returned query.Component object.</p> <p>Use the method query.create(options) to create your initial query definition (query.Query). The initial query definition uses one query type. For available query types, see query.Type.</p> <p>After you create the initial query definition, use this method to create explicit directional joins from other components to this component.</p> <div style="border: 1px solid #f0e68c; padding: 5px; margin-top: 10px;"> Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types. </div>
Returns	query.Component
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fieldId	string	required	<p>The column type (field type) that joins the parent component to the new component.</p> <p>Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser.</p>
options.source	string	required	<p>The query type of the component joined to this component. This value sets the Component.source property.</p> <p>This value can be described as the inverse relationship of this component, and it determines the source query type of the newly joined component.</p>

Errors

Error Code	Thrown If
RELATIONSHIP_ALREADY_USED	The specified join relationship already exists.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myEmployeeQuery = query.create({
4   type: query.Type.EMPLOYEE
5 });
6
7 var mySalesOrderJoin = myEmployeeQuery.joinFrom({
8   fieldId: 'salesrep',
9   source: 'salesorder'
10 });
11
12 var items = mySalesOrderJoin.autoJoin({
13   fieldId: 'item'
14 });
15
16 myEmployeeQuery.columns = [
17   myEmployeeQuery.createColumn({
18     fieldId: 'entityid'
19   }),
20   myEmployeeQuery.createColumn({
21     fieldId: 'hiredate'
22   }),
23   mySalesOrderJoin.createColumn({
24     fieldId: 'id'
25   }),
26   mySalesOrderJoin.createColumn({
27     fieldId: 'trandate'
28   })
29 ];
30
31 var firstSort = myEmployeeQuery.createSort({
32   column: myEmployeeQuery.columns[0],
33   ascending: false
34 });
35 var secondSort = myEmployeeQuery.createSort({
36   column: myEmployeeQuery.columns[1],
37   ascending: true
38 });
39 myEmployeeQuery.sort = [firstSort, secondSort];
40
41 var results = myEmployeeQuery.run();
42 ...
43 // Add additional code

```

Component.joinTo(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates an explicit directional join relationship to another component from this component (a polymorphic join). This method sets the Component.target property on the returned query.Component object.
--------------------	---

	<p>Use the method query.create(options) to create your initial query definition (<code>query.Query</code>). The initial query definition uses one query type. For available query types, see query.Type.</p> <p>After you create the initial query definition, use this method to create explicit directional joins to other components from this component.</p> <div style="border: 1px solid #f0e68c; padding: 5px; margin-top: 10px;"> <p> Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types.</p> </div>
Returns	<code>query.Component</code>
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	<code>query.Component</code>
Sibling Object Members	Component Object Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.fieldId</code>	string	required	<p>The column type (field type) that joins the parent component to the new component.</p> <p>Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser.</p>
<code>options.target</code>	string	required	<p>The query type of the component joined to this component. This value sets the Component.target property.</p> <p>This value can be described as the polymorphic relationship of this component, and it determines the target query type of the newly joined component.</p>

Errors

Error Code	Thrown If
<code>RELATIONSHIP_ALREADY_USED</code>	The specified join relationship already exists.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEntityJoin = myTransactionQuery.joinTo({
8   fieldId: 'entity',
9   target: query.Type.CUSTOMER
10 });
11
12 myTransactionQuery.columns = [
13   myEntityJoin.createColumn({
14     fieldId: 'subsidiary'
15   })
16 ];
17
18 myTransactionQuery.sort = [
19   myTransactionQuery.createSort({
20     column: myTransactionQuery.columns[0],
21     ascending: false
22   })
23 ];
24
25 var results = myTransactionQuery.runPaged({
26   pageSize: 10
27 });
28 ...
29 // Add additional code

```

Component.child



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A reference to children of this component. The value of this property is an object of key/value pairs. Each key is the name of a child component. Each respective value refers to the corresponding query.Component object. The object values are set during the execution of Query.autoJoin(options) and Component.autoJoin(options) . The order of the key/value pairs reflects the parent/child hierarchy.
Type	Object (read-only)
Module	N/query Module
Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myDeptJoin = mySalesRepJoin.autoJoin({
12   fieldId: 'department'
13 });
14
15 var theChild = mySalesRepJoin.child;
16 ...
17 // Add additional code

```

Component.parent



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A reference to the parent query.Component object of this component. This property is set during the execution of Query.autoJoin(options) or Component.autoJoin(options) .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myDeptJoin = mySalesRepJoin.autoJoin({
12   fieldId: 'department'
13 });

```

```

14 |
15 var theParent = myDeptJoin.parent;
16 ...
17 // Add additional code

```

Component.source



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The query type of the component joined to this component. This property can also be described as the inverse relationship of this component. This property is set during the execution of Query.joinFrom(options) and Component.joinFrom(options) .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myEmployeeQuery = query.create({
4   type: query.Type.EMPLOYEE
5 });
6
7 var myTransactionJoin = myEmployeeQuery.joinFrom({
8   fieldId: 'entity',
9   source: 'transaction'
10 });
11
12 var theSource = myTransactionJoin.source;
13 ...
14 // Add additional code

```

Component.target



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The query type of this component. This property can also be described as the polymorphic relationship of this component. This property is set during the execution of Query.joinTo(options) and Component.joinTo(options) .
Type	string (read-only)
Module	N/query Module

Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEmployeeJoin = myTransactionQuery.joinTo({
8   fieldId: 'createdby',
9   target: 'employee'
10 });
11
12 var theTarget = myEmployeeJoin.target;
13 ...
14 // Add additional code

```

Component.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The query type of this component. This property is set during the execution of Query.autoJoin(options) and Component.autoJoin(options) .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Component
Sibling Object Members	Component Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var theType = myCustomerQuery.type;
8 ...

```

```
9 // Add additional code
```

query.Condition



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A condition that narrows the query results. The <code>query.Condition</code> object acts in the same capacity as the <code>search.Filter</code> object in the N/search Module . The primary difference is that <code>query.Condition</code> objects can contain other <code>query.Condition</code> objects.
	To create conditions:
	<ul style="list-style-type: none"> ■ Use <code>Query.createCondition(options)</code> to create conditions for the initial query definition created with <code>query.create(options)</code>. ■ Use <code>Component.createCondition(options)</code> to create conditions for the join relationships created with <code>Query.autoJoin(options)</code> and <code>Component.autoJoin(options)</code>. ■ If you have multiple conditions, use them to create a new nested condition with the methods <code>Query.and()</code>, <code>Query.or()</code>, and <code>Query.not()</code>. ■ Assign your simple or nested condition to <code>Query.condition</code>. For an example, see Syntax.
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	Condition Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myLocationJoin = mySalesRepJoin.autoJoin({
12   fieldId: 'location'
13 });
14
15 var firstCondition = myCustomerQuery.createCondition({
16   fieldId: 'id',
17   operator: query.Operator.EQUAL,
18   values: 107
19 });
20 var secondCondition = myCustomerQuery.createCondition({
21   fieldId: 'id',
22   operator: query.Operator.EQUAL,
23   values: 2647
24 });

```

```

25 var thirdCondition = mySalesRepJoin.createCondition({
26   fieldId: 'email',
27   operator: query.Operator.START_WITH_NOT,
28   values: 'foo'
29 });
30
31 myCustomerQuery.condition = myCustomerQuery.and(
32   thirdCondition, myCustomerQuery.not(
33     myCustomerQuery.or(firstCondition, secondCondition)
34   )
35 );
36
37 var resultSet = myCustomerQuery.run();
38 ...
39 // Add additional code

```

Condition.aggregate

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An aggregate function that is performed on the condition. An aggregate function performs a calculation on the condition values and returns a single value. This property is set during the execution of Query.createCondition(options) or Component.createCondition(options) .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Condition
Sibling Object Members	Condition Object Members
Since	2018.1

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var myAggregateCondition = myCustomerQuery.createCondition({
8   fieldId: 'openingbalance',
9   operator: query.Operator.GREATER,
10  values: 10000,
11  aggregate: query.Aggregate.MAXIMUM
12 });
13
14 var theAggregate = myAggregateCondition.aggregate;
15 ...
16 // Add additional code

```

Condition.children



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of child conditions used to create the parent condition.
	 Note: This property is applicable to only parent conditions created with the execution of Query.and() , Query.or() , or Query.not() .
Type	query.Condition[] (read-only)
Module	N/query Module
Parent Object	query.Condition
Sibling Object Members	Condition Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var myFirstCondition = myCustomerQuery.createCondition({
8   fieldId: 'openingbalance',
9   operator: query.Operator.GREATER,
10  values: 10000
11 });
12
13 var mySecondCondition = myCustomerQuery.createCondition({
14   fieldId: 'email',
15   operator: query.Operator.START_WITH_NOT,
16   values: 'foo'
17 });
18
19 var myComplexCondition = myCustomerQuery.and(myFirstCondition, mySecondCondition);
20
21 var theChildren = myComplexCondition.children;
22 ...
23 // Add additional code

```

Condition.component



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The component used to created the condition This property is set during the execution of Query.createCondition(options) and Component.createCondition(options) .
-----------------------------	---

	Note: This property is not applicable to parent conditions created with the execution of <code>Query.and()</code> , <code>Query.or()</code> , or <code>Query.not()</code> .
Type	<code>query.Component</code> (read-only)
Module	N/query Module
Parent Object	<code>query.Condition</code>
Sibling Object Members	Condition Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myCondition = mySalesRepJoin.createCondition({
12   fieldId: 'email',
13   operator: query.Operator.START_WITH,
14   values: 'mentor'
15 });
16
17 var theComponent = myCondition.component;
18 ...
19 // Add additional code

```

Condition.fieldId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The name of the condition.</p> <p>This property is set during the execution of <code>Query.createCondition(options)</code> and <code>Component.createCondition(options)</code>.</p> <p>Note: This property is not applicable to parent conditions created with the execution of <code>Query.and()</code>, <code>Query.or()</code>, or <code>Query.not()</code>.</p>
Type	<code>string</code> (read-only)
Module	N/query Module
Parent Object	<code>query.Condition</code>
Sibling Object Members	Condition Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var myCondition = myCustomerQuery.createCondition({
8   fieldId: 'openingbalance',
9   operator: query.Operator.GREATER,
10  values: 10000
11 });
12
13 var theFieldId = myCondition.fieldId;
14 ...
15 // Add additional code

```

Condition.formula



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The formula used to create the condition. This property is set during the execution of Query.createCondition(options) and Component.createCondition(options) . For more information on formulas, see the help topics Formulas in Search and SQL Expressions .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Condition
Sibling Object Members	Condition Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myFormulaCondition = myTransactionQuery.createCondition({
8   formula: '{amount} * 125',

```

```

9   operator: query.Operator.GREATER,
10  values: 50000,
11  type: query.ReturnType.CURRENCY
12 });
13
14 var theFormula = myFormulaCondition.formula;
15 ...
16 //Add additional code

```

Condition.operator



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of the operator used to create the condition. This property is set during the execution of Query.createCondition(options) and Component.createCondition(options) . <div style="background-color: #e0f2ff; padding: 10px;"> i Note: This property is not applicable to parent conditions created with the execution of Query.and(), Query.or(), or Query.not(). </div>
Type	string (read-only)
Module	N/query Module
Parent Object	query.Condition
Sibling Object Members	Condition Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var myCondition = myCustomerQuery.createCondition({
8   fieldId: 'openingbalance',
9   operator: query.Operator.GREATER,
10  values: 10000
11 });
12
13 var theOperator = myCondition.operator;
14 ...
15 // Add additional code

```

Condition.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The return type of the formula used to create the condition.
-----------------------------	--

	<p>This property is set during the execution of Query.createCondition(options) or Component.createCondition(options).</p> <p>For more information on formulas, see the help topics Formulas in Search and SQL Expressions.</p> <p>Note: This property is not applicable to parent conditions created with the execution of Query.and(), Query.or(), or Query.not().</p>
Type	string (read-only)
Module	N/query Module
Parent Object	query.Condition
Sibling Object Members	Condition Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myFormulaCondition = myTransactionQuery.createCondition({
8   formula: '{amount} * 125',
9   operator: query.Operator.GREATER,
10  values: 50000,
11  type: query.ReturnType.CURRENCY
12 });
13
14 var theFormulaType = myFormulaCondition.type;
15 ...
16 // Add additional code

```

Condition.values

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>An array of values used by an operator to create the condition.</p> <p>This property is set by passing in values for options.fieldId, options.operator and options.values during the execution of Query.createCondition(options) or Component.createCondition(options).</p> <p>Note: This property is not applicable to parent conditions created with the execution of Query.and(), Query.or(), or Query.not().</p>
Type	string[] number[] boolean[] Date[] query.RelativeDate[] query.Period (read-only)
Module	N/query Module
Parent Object	query.Condition

Sibling Object Members	Condition Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var myCondition = myCustomerQuery.createCondition({
8   fieldId: 'firstname',
9   operator: query.Operator.ANY_OF,
10  values: ['Martin', 'Russell', 'Janina']
11 });
12
13 var theValues = myCondition.values;
14 ...
15 // Add additional code

```

query.Page

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	One page of the paged query results.
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	Page Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10}),
11  myCustomerQuery.createColumn({
12    fieldId: 'firstname'
13})
14];

```

```

13   }),
14   myCustomerQuery.createColumn({
15     fieldId: 'email'
16   })
17 ];
18
19 var myPagedResults = myCustomerQuery.runPaged({
20   pageSize: 10
21 });
22
23 // Fetch results using an iterator
24 var iterator = myPagedResults.iterator();
25 iterator.each(function(resultPage) {
26   var currentPage = resultPage.value;
27   log.debug(currentPage.pageRange.size);
28   return true;
29 });
30
31 // Alternatively, fetch results using a loop
32 for (var i = 0; i < myPagedResults.pageRanges.length; i++) {
33   var currentPage = myPagedResults.fetch(i);
34   log.debug(currentPage.pageRange.size);
35 }
36 ...
37 // Add additional code

```

Page.data

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The query results contained in this page.
Type	query.ResultSet (read-only)
Module	N/query Module
Parent Object	query.Page
Sibling Object Members	Page Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 var myPagedResults = myCustomerQuery.runPaged({
14   pageSize: 10
15 });
16
17 var iterator = myPagedResults.iterator();

```

```

18 | iterator.each(function(resultPage) {
19 |   var currentPage = resultPage.value;
20 |   var theData = currentPage.data;
21 |   return true;
22 | });
23 |
24 | // Add additional code

```

Page.isFirst



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether the page is the first of the paged query results.
Type	boolean (read-only)
Module	N/query Module
Parent Object	query.Page
Sibling Object Members	Page Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 | // Add additional code
2 |
3 | var myCustomerQuery = query.create({
4 |   type: query.Type.CUSTOMER
5 | });
6 |
7 | myCustomerQuery.columns = [
8 |   myCustomerQuery.createColumn({
9 |     fieldId: 'entityid'
10 |   })
11 | ];
12 |
13 | var myPagedResults = myCustomerQuery.runPaged({
14 |   pageSize: 10
15 | });
16 |
17 | var iterator = myPagedResults.iterator();
18 | iterator.each(function(resultPage) {
19 |   var currentPage = resultPage.value;
20 |   var isFirst = currentPage.isFirst;
21 |   return true;
22 | });
23 |
24 | // Add additional code

```

Page.isLast



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether the page is the last of the paged query results.
-----------------------------	--

Type	boolean (read-only)
Module	N/query Module
Parent Object	query.Page
Sibling Object Members	Page Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10 }),
11   myCustomerQuery.createColumn({
12     fieldId: 'firstname'
13 }),
14   myCustomerQuery.createColumn({
15     fieldId: 'email'
16 })
17 ];
18
19 var myPagedResults = myCustomerQuery.runPaged({
20   pageSize: 10
21 });
22
23 var iterator = myPagedResults.iterator();
24 iterator.each(function(resultPage) {
25   var currentPage = resultPage.value;
26   var isLast = currentPage.isLast;
27   return true;
28 });
29 ...
30 // Add additional code

```

Page.pageRange



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The range of query results for this page.
Type	query.PageRange (read-only)
Module	N/query Module
Parent Object	query.Page
Sibling Object Members	Page Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 var myPagedResults = myCustomerQuery.runPaged({
14   pageSize: 10
15 });
16
17 var iterator = myPagedResults.iterator();
18 iterator.each(function(resultPage) {
19   var currentPage = resultPage.value;
20   var thePageRange = currentPage.pageRange;
21   return true;
22 });
23 ...
24 // Add additional code

```

Page.pagedData



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The set of paged query results that this page is from.
Type	query.PagedData (read-only)
Module	N/query Module
Parent Object	query.Page
Sibling Object Members	Page Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];

```

```

12 | var myPagedResults = myCustomerQuery.runPaged({
13 |   pageSize: 10
14 | });
15 |
16 |
17 | var iterator = myPagedResults.iterator();
18 | iterator.each(function(resultPage) {
19 |   var currentPage = resultPage.value;
20 |   var thePagedData = currentPage.pagedData;
21 |   return true;
22 | });
23 | ...
24 | // Add additional code

```

query.PagedData



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A set of paged query results. This object also contains information about the set of paged results it encapsulates. Use Query.runPaged() or Query.runPaged.promise() to create this object. For paged queries, the maximum number of result rows per page is 1000. The minimum number of result rows per page is 5, except for the last page in the result set (because the last page may include fewer than 5 results).
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	PagedData Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   }),
11   myCustomerQuery.createColumn({
12     fieldId: 'firstname'
13   }),
14   myCustomerQuery.createColumn({
15     fieldId: 'email'
16   })
17 ];
18
19 var myPagedResults = myCustomerQuery.runPaged({
20   pageSize: 10

```

```

21 });
22
23 // Fetch results using an iterator
24 var iterator = myPagedResults.iterator();
25 iterator.each(function(resultPage) {
26   var currentPage = resultPage.value;
27   var currentPagedData = currentPage.pagedData;
28   log.debug(currentPage.pageRange.size);
29   return true;
30 });
31
32 // Alternatively, fetch results using a loop
33 for (var i = 0; i < myPagedResults.pageRanges.length; i++) {
34   var currentPage = myPagedResults.fetch(i);
35   var currentPagedData = currentPage.pagedData;
36   log.debug(currentPage.pageRange.size);
37 }
38 ...
39 // Add additional code

```

PagedData.iterator()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Standard SuiteScript 2.0 object for iterating through results
Returns	Iterator object
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.PagedData
Sibling Object Members	PagedData Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 var myPagedResults = myCustomerQuery.runPaged({
14   pageSize: 10
15 });
16
17 var iterator = myPagedResults.iterator();

```

```

18 iterator.each(function(resultPage) {
19     var currentPage = resultPage.value;
20     var currentPagedData = currentPage.pagedData;
21     return true;
22 });
23 ...
24 // Add additional code

```

PagedData.count

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The total number of paged query result rows.
Type	number (read-only)
Module	N/query Module
Parent Object	query.PagedData
Sibling Object Members	PagedData Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4     type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8     myCustomerQuery.createColumn({
9         fieldId: 'entityid'
10    })
11];
12
13 var myPagedResults = myCustomerQuery.runPaged({
14     pageSize: 10
15 });
16
17 var iterator = myPagedResults.iterator();
18 iterator.each(function(resultPage) {
19     var currentPage = resultPage.value;
20     var currentPagedData = currentPage.pagedData;
21     var theCount = currentPagedData.count;
22     return true;
23 });
24 ...
25 // Add additional code

```

PagedData.pageRanges

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of page ranges for the paged query results.
-----------------------------	--

Type	query.PageRange[]
Module	N/query Module
Parent Object	query.PagedData
Sibling Object Members	PagedData Object Members
Since	2018.1

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 var myPagedResults = myCustomerQuery.runPaged({
14   pageSize: 10
15 });
16
17 var iterator = myPagedResults.iterator();
18 iterator.each(function(resultPage) {
19   var currentPage = resultPage.value;
20   var currentPagedData = currentPage.pagedData;
21   var thePageRanges = currentPagedData.pageRanges;
22   return true;
23 });
24 ...
25 // Add additional code

```

PagedData.pageSize

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The number of query result rows per page. For paged queries, the maximum number of result rows per page is 1000. The minimum number of result rows per page is 5, except for the last page in the result set (because the last page may include fewer than 5 results).
Type	number (read-only)
Module	N/query Module
Parent Object	query.PagedData
Sibling Object Members	PagedData Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 var myPagedResults = myCustomerQuery.runPaged({
14   pageSize: 10
15 });
16
17 var iterator = myPagedResults.iterator();
18 iterator.each(function(resultPage) {
19   var currentPage = resultPage.value;
20   var currentPagedData = currentPage.pagedData;
21   var thePageSize = currentPagedData.pageSize;
22   return true;
23 });
24
25 // Add additional code

```

query.PageRange

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The range of query results for a page.
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	PageRange Object Members
Since	2018.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [

```

```

8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10    }),
11   myCustomerQuery.createColumn({
12     fieldId: 'firstname'
13    }),
14   myCustomerQuery.createColumn({
15     fieldId: 'email'
16    })
17  ];
18
19 var myPagedResults = myCustomerQuery.runPaged({
20   pageSize: 10
21 });
22
23 // Fetch results using an iterator
24 var iterator = myPagedResults.iterator();
25 iterator.each(function(resultPage) {
26   var currentPage = resultPage.value;
27   var currentPageRange = currentPage.pageRange;
28   log.debug(currentPageRange.size);
29   return true;
30 });
31
32 // Alternatively, fetch results using a loop
33 for (var i = 0; i < myPagedResults.pageRanges.length; i++) {
34   var currentPage = myPagedResults.fetch(i);
35   var currentPageRange = currentPage.pageRange;
36   log.debug(currentPageRange.size);
37 }
38 ...
39 // Add additional code

```

PageRange.index



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The array index for this page range.
Type	number (read-only)
Module	N/query Module
Parent Object	query.PageRange
Sibling Object Members	PageRange Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'

```

```

10    });
11  ];
12
13 var myPagedResults = myCustomerQuery.runPaged({
14   pageSize: 10
15 });
16
17 var iterator = myPagedResults.iterator();
18 iterator.each(function(resultPage) {
19   var currentPage = resultPage.value;
20   var currentPageRange = currentPage.pageRange;
21   var theIndex = currentPageRange.index;
22   return true;
23 });
24 ...
25 // Add additional code

```

PageRange.size

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The number of query result rows in this page range.
Type	number (read-only)
Module	N/query Module
Parent Object	query.PageRange
Sibling Object Members	PageRange Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 var myPagedResults = myCustomerQuery.runPaged({
14   pageSize: 10
15 });
16
17 var iterator = myPagedResults.iterator();
18 iterator.each(function(resultPage) {
19   var currentPage = resultPage.value;
20   var currentPageRange = currentPage.pageRange;
21   var theSize = currentPageRange.size;
22   return true;
23 });
24 ...
25 // Add additional code

```

query.Period



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A period of time to use in query conditions. Use query.createPeriod(options) to create this object. After you create this object, you can use it in the values parameter of Query.createCondition(options) or Component.createCondition(options) .
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	Period Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPeriod = query.createPeriod({
4   code: query.PeriodCode.LAST_PERIOD,
5   adjustment: query.PeriodAdjustment.ALL,
6   type: query.PeriodType.END
7 });
8
9 // myQuery is an existing query.Query object
10 var myComplexCondition = myQuery.createCondition({
11   fieldId: 'trandate',
12   operator: query.Operator.BEFORE,
13   values: [myPeriod]
14 });
15 ...
16 // Add additional code

```

Period.adjustment



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The adjustment of the period. This property uses values from the query.PeriodAdjustment enum. If you create a period using query.createPeriod(options) and do not specify a value for the options.adjustment parameter, the default value of this property is query.PeriodAdjustment.NOT_LAST .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Period

Sibling Object Members	Period Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPeriod = query.createPeriod({
4   code: query.PeriodCode.LAST_PERIOD,
5   adjustment: query.PeriodAdjustment.ALL,
6   type: query.PeriodType.END
7 });
8
9 var theAdjustment = myPeriod.adjustment;
10 ...
11 // Add additional code

```

Period.code



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The code of the period. This property uses values from the query.PeriodCode enum.
Type	string (read-only)
Module	N/query Module
Parent Object	query.Period
Sibling Object Members	Period Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPeriod = query.createPeriod({
4   code: query.PeriodCode.LAST_PERIOD,
5   adjustment: query.PeriodAdjustment.ALL,
6   type: query.PeriodType.END
7 });
8
9 var theCode = myPeriod.code;
10 ...
11 // Add additional code

```

Period.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The type of the period.
	This property uses values from the query.PeriodType enum. If you create a period using query.createPeriod(options) and do not specify a value for the options.type parameter, the default value of this property is query.PeriodType.START.
Type	string (read-only)
Module	N/query Module
Parent Object	query.Period
Sibling Object Members	Period Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPeriod = query.createPeriod({
4   code: query.PeriodCode.LAST_PERIOD,
5   adjustment: query.PeriodAdjustment.ALL,
6   type: query.PeriodType.END
7 });
8
9 var theType = myPeriod.type;
10 ...
11 // Add additional code

```

query.Query



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>The query.Query object encapsulates the query definition. To create a query with the N/query module:</p> <ol style="list-style-type: none"> 1. Use the query.create(options) method to create your query definition (this object). The initial query definition uses one query type. For available query types, see query.Type. 2. After you create the initial query definition, use Query.autoJoin(options) to create your first join. 3. Then use either Query.autoJoin(options) or Component.autoJoin(options) to create subsequent joins. <p>The query definition always contains at least one query.Component object. The query.Component object encapsulates one component of the query definition. Each new component is created as a child to the previous component, and all components exist as children to the query definition.</p>
---------------------------	--

	<p>You can think of a component as a building block; each new component builds on the previous component created. The last component created encapsulates the relationship between it and all of its parent components.</p> <p>Queries with joins contain multiple components. The query definition contains a child <code>query.Component</code> object for each of the following:</p> <ul style="list-style-type: none"> ■ The initial query definition: The initial <code>query.Component</code> object is called the root component. It encapsulates the initial query type passed to <code>query.create(options)</code>. The root component is automatically created with the initial query definition and is a child to the <code>query.Query</code> object. The <code>Query.root</code> property contains a reference to the root component. ■ The first join: The second <code>query.Component</code> object is created with <code>Query.autoJoin(options)</code>. It encapsulates the relationship between the initial query definition and the second query type. This relationship is determined by the field ID passed to <code>Query.autoJoin(options)</code>. The second <code>query.Component</code> object is a child to the root component. ■ Each subsequent join: The third <code>query.Component</code> object is created with <code>Query.autoJoin(options)</code> or <code>Component.autoJoin(options)</code>. All subsequent joins are also created with <code>Query.autoJoin(options)</code> or <code>Component.autoJoin(options)</code>. Each of these <code>query.Component</code> objects encapsulates the relationship between all previous query types and the new query type. This relationship is determined by the field ID passed to <code>Component.autoJoin(options)</code>.
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	Query Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEntityJoin = myTransactionQuery.autoJoin({
8   fieldId: 'entity'
9 });
10
11 myTransactionQuery.columns = [
12   myEntityJoin.createColumn({
13     fieldId: 'subsidiary'
14 })];
15
16 myTransactionQuery.sort = [
17   myTransactionQuery.createSort({
18     column: myTransactionQuery.columns[0],
19     ascending: false
20   })
21 ];
22
23 var results = myTransactionQuery.runPaged({
24   pageSize: 10
25 });
26 ...

```

```
27 // Add additional code
```

Query.and()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a new condition (a query.Condition object) that corresponds to a logical conjunction (AND) of the arguments passed to the method. The arguments must be one or more query.Condition objects.</p> <p>A condition narrows the query results. The query.Condition object acts in the same capacity as the search.Filter object in the N/search Module. The primary difference is that query.Condition objects can contain other query.Condition objects.</p> <p>To create conditions:</p> <ul style="list-style-type: none"> ■ Use Query.createCondition(options) to create conditions for the initial query definition created with query.create(options). ■ Use Component.createCondition(options) to create conditions for the join relationships created with Query.autoJoin(options) and Component.autoJoin(options). ■ If you have multiple conditions, use them to create a new parent condition with the methods Query.and(), Query.or(), and Query.not(). ■ Assign your parent condition to Query.condition. For an example, see Syntax.
Returns	query.Condition
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Parameters

Parameter	Type	Required / Optional	Description
condition1 — n	query.Condition[]	required	<p>One or more condition objects.</p> <p>There is no limit on the number of conditions you can specify.</p>

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```
1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 })
```

```

5  });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myLocationJoin = mySalesRepJoin.autoJoin({
12   fieldId: 'location'
13 });
14
15 var firstCondition = myCustomerQuery.createCondition({
16   fieldId: 'id',
17   operator: query.Operator.EQUAL,
18   values: 107
19 });
20 var secondCondition = myCustomerQuery.createCondition({
21   fieldId: 'id',
22   operator: query.Operator.EQUAL,
23   values: 2647
24 });
25 var thirdCondition = mySalesRepJoin.createCondition({
26   fieldId: 'email',
27   operator: query.Operator.START_WITH_NOT,
28   values: 'foo'
29 });
30
31 myCustomerQuery.condition = myCustomerQuery.and(
32   thirdCondition, myCustomerQuery.not(
33     myCustomerQuery.or(firstCondition, secondCondition)
34   )
35 );
36
37 var resultSet = myCustomerQuery.run();
38 ...
39 // Add additional code

```

Query.autoJoin(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a join relationship.</p> <p>Use the method query.create(options) to create your initial query definition (query.Query). The initial query definition uses one query type. For available query types, see query.Type.</p> <p>After you create the initial query definition, use <code>Query.autoJoin(options)</code> to create your first join (query.Component). Then use Component.autoJoin(options) to create each subsequent join (query.Component).</p> <div data-bbox="479 1453 1380 1564" style="border: 1px solid #ccc; padding: 10px;"> <p>Note: This method is a shortcut for the chained <code>Query.root</code> and <code>Component.autoJoin(options)</code>: <code>Query.root.join(options)</code>. The <code>Query.root</code> property references the root component, which is a query.Component object.</p> </div> <div data-bbox="479 1592 1380 1704" style="border: 1px solid #ccc; padding: 10px;"> <p>Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types.</p> </div>
Returns	query.Component
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fieldId	string	required	<p>The column type (field type) that joins the parent component to the new component. This value determines the columns on which the components are joined and the type of the newly joined component.</p> <p>Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser.</p>

Errors

Error Code	Thrown If
RELATIONSHIP_ALREADY_USED	The specified join relationship already exists.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEntityJoin = myTransactionQuery.autoJoin({
8   fieldId: 'entity'
9 });
10
11 myTransactionQuery.columns = [
12   myEntityJoin.createColumn({
13     fieldId: 'subsidiary'
14   })
15 ];
16
17 myTransactionQuery.sort = [
18   myTransactionQuery.createSort({
19     column: myTransactionQuery.columns[0],
20     ascending: false
21   })
22 ];
23
24 var results = myTransactionQuery.runPaged({

```

```

25     pageSize: 10
26   });
27   ...
28 // Add additional code

```

Query.createColumn(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a query result column based on the query.Query object. The query.Column object is the equivalent of the search.Column object in the N/search Module. The query.Column object describes the field types (columns) that are displayed from the query results.</p> <p>To create columns:</p> <ul style="list-style-type: none"> ■ Use <code>Query.createColumn(options)</code> to create columns on the initial query definition created with query.create(options). Use this method in one of two ways: <ul style="list-style-type: none"> □ Pass in an argument for the parameter <code>options.fieldId</code>. □ Pass in an argument for the parameter <code>options.formula</code>. If you use this option, you can also use the optional parameter <code>options.type</code>. ■ If needed, use Component.createColumn(options) to create conditions on the join relationships created with Query.autoJoin(options) and Component.autoJoin(options). ■ Assign all created columns as array values to Query.columns. For an example, see Syntax. <p>When you create a column, you can specify a field context. The field context determines how field values are displayed in the column. For example, you can specify that a column should display raw data (such as internal IDs), consolidated or converted amounts (such as currency totals), or user-friendly values (such as names). You can specify a field context in two ways:</p> <ul style="list-style-type: none"> ■ Use a context from the query.FieldContext enum directly as the value of the <code>options.context</code> parameter. For example: <pre> 1 myTransactionLine.createColumn({ 2 fieldId: 'netamount', 3 context: query.FieldContext.CURRENCY_CONOLIDATED 4 }); </pre> <p>This example is the simplest way to specify a field context that does not accept additional parameters. Because the <code>options.context</code> parameter is an Object, this example is equivalent to the following:</p> <pre> 1 myTransactionLine.createColumn({ 2 fieldId: 'netamount', 3 context: { 4 name: query.FieldContext.CURRENCY_CONOLIDATED 5 } 6 }); </pre> <ul style="list-style-type: none"> ■ Use a context from the query.FieldContext enum as the value of the <code>options.context.name</code> parameter, and specify additional parameters using the <code>options.context.params</code> parameter. For example: <pre> 1 myTransactionLine.createColumn({ 2 fieldId: 'netamount', 3 context: { 4 name: query.FieldContext.CONVERTED, 5 params: { 6 currencyId: 4, 7 date: new Date('2019/01/01') 8 } 9 } 10 }); </pre>
---------------------------	--

	<p>In this example, the created column displays the value of the netamount currency field using the exchange rate that was in effect on January 1, 2019 for the currency with an ID of 4.</p> <p>In this release, only the query.FieldContext.CONVERTED context uses additional parameters. The supported parameters are currencyId and date. For the date parameter, you can pass a JavaScript Date object or query.RelativeDate object. If you pass a query.RelativeDate object using a value from the query.RelativeDateRange enum, use the start property or end property to specify the exact date of the exchange rate. For example, to use the exchange rate that was in effect at the beginning of the last fiscal quarter:</p> <pre> 1 myTransactionLine.createColumn({ 2 fieldId: 'netamount', 3 context: { 4 name: query.FieldContext.CONVERTED, 5 params: { 6 currencyId: 4, 7 date: query.RelativeDateRange.LAST_FISCAL_QUARTER.start 8 } 9 } 10 }); </pre> <p>If you use only the query.RelativeDate object from the query.RelativeDateRange enum and do not specify either the start or end properties, the end date of the relative date range is used. This behavior means that the following two date properties are equivalent:</p> <ul style="list-style-type: none"> ■ date: query.RelativeDateRange.LAST_FISCAL_QUARTER ■ date: query.RelativeDateRange.LAST_FISCAL_QUARTER.end <p>Note: This method is a shortcut for the chained <code>Query.root</code> and <code>Component.createColumn(options)</code>: <code>Query.root.createColumn(options)</code>. The <code>Query.root</code> property references the root component, which is a <code>query.Component</code> object.</p>
Returns	<code>query.Column</code>
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Parameters

Note:	The options parameter is a JavaScript object.
--------------	---

Parameter	Type	Required / Optional	Description
<code>options.fieldId</code>	string	required if <code>options.formula</code> is not used	The field ID of the query result column. This value sets the <code>Column.fieldId</code> property. Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and

Parameter	Type	Required / Optional	Description
			the N/query module. For more information, see the help topic Analytics Browser .
options.formula	string	required if options.fieldId is not used	<p>The formula used to create the query result column. This value sets the Column.formula property.</p> <p>For more information on formulas, see the help topics Formulas in Search and SQL Expressions.</p>
options.type	string	required if options.formula is used	<p>If you use the options.formula parameter, use this parameter to explicitly define the formula's return type. This value sets the Column.type property.</p> <p>Use the appropriate query.ReturnType enum value to pass in your argument. This enum holds all the supported values for this parameter.</p>
options.aggregate	string	optional	<p>Use this parameter to run an aggregate function on your query result column. An aggregate function performs a calculation on the column values and returns a single value. This value sets the Column.aggregate property.</p> <p>Use the appropriate query.Aggregate enum value to pass in your argument. This enum holds all the supported values for this parameter.</p>
options.alias	string	optional	<p>The alias for the column. An alias is an alternate name for a column, and the alias is used in mapped results. This value sets the Column.alias property.</p> <p>You must specify an alias in certain situations if you want to use ResultSet.asMappedResults() or Result.asMap(). For more information, see Column.alias.</p>
options.groupBy	boolean	optional	<p>Indicates whether the query results are grouped by this query result column. This value sets the Column.groupBy property.</p> <p>If you do not pass in an argument, the default value is set to false.</p>
options.label	string	optional	<p>The label for the column.</p> <p>A label is important if the query object is used as the data source for printing (for example, in the TemplateRenderer.addQuery(options) method in the N/render module).</p>
options.context	Object	optional	<p>The field context for values in the query result column. This value sets the Column.context property.</p> <p>If you do not pass in an argument, the default value is set to query.FieldContext.RAW.</p>
options.context.name	string	required if options.context is used	<p>The name of the field context.</p> <p>Use the appropriate query.FieldContext enum value to pass in your argument. This enum holds all the supported values for this parameter.</p>
options.context.params	Object	required if options.context.name has a value of query.FieldContext.CONVERTED	<p>The additional parameters to use with the specified field context.</p> <p>In this release, only the query.FieldContext.CONVERTED context uses</p>

Parameter	Type	Required / Optional	Description
			additional parameters. The supported parameters are currencyId and date.
options.context.params.currencyId	number	required if options.context.name has a value of query.FieldContext.CONVERTED	The ID of the currency to convert to.
options.context.params.date	query.RelativeDate JavaScript Date	required if options.context.name has a value of query.FieldContext.CONVERTED	The date to use for the actual exchange rate between the base currency and the currency to convert to. For example, if you want to use the exchange rate that was in effect on March 3, 2019, specify a query.RelativeDate object or JavaScript Date object that represents this date.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   myCustomerQuery.createColumn({
16     fieldId: 'id'
17   }),
18   mySalesRepJoin.createColumn({
19     fieldId: 'entityid'
20   }),
21   mySalesRepJoin.createColumn({
22     fieldId: 'email'
23   }),
24   mySalesRepJoin.createColumn({
25     fieldId: 'hiredate'
26   })
27 ];
28
29 myCustomerQuery.sort = [
30   myCustomerQuery.createSort({
31     column: myCustomerQuery.columns[1]
32   }),
33   mySalesRepJoin.createSort({
34     column: mySalesRepJoin.columns[0],
35     ascending: false
36   })
37 ];
38
39 var resultSet = myCustomerQuery.run();
40 ...
41 // Add additional code

```

Query.createCondition(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a condition (query filter) based on the query.Query object. A condition narrows the query results. The query.Condition object acts in the same capacity as the search.Filter object in the N/search Module. The primary difference is that query.Condition objects can contain other query.Condition objects.</p> <p>To create conditions:</p> <ul style="list-style-type: none"> ■ Use <code>Query.createCondition(options)</code> to create conditions on the initial query definition created with query.create(options). Use this method in one of two ways: <ul style="list-style-type: none"> □ Pass in arguments for the parameters <code>options.fieldId</code>, <code>options.operator</code>, and <code>options.values</code>. The combination of these arguments translates to <code><filter column><operator><field value></code> (for example, 'city' equals 'Boston'). □ Pass in an argument for the parameter <code>options.formula</code>. If you use this option, you can also use the optional parameter <code>options.type</code>. ■ If needed, use Component.createCondition(options) to create conditions on the join relationships created with Query.autoJoin(options) and Component.autoJoin(options). ■ If you have multiple conditions, use them to create a new nested condition with the methods Query.and(), Query.or(), and Query.not(). ■ Assign your simple or nested condition to Query.condition. For an example, see Syntax. <p>Note: This method is a shortcut for the chained Query.root and Component.createCondition(options): <code>Query.root.createCondition(options)</code>. The Query.root property references the root component, which is a query.Component object.</p>
Returns	query.Condition
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Parameters



Note: The `options` parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.fieldId</code>	string	required if <code>options.operator</code> and <code>options.values</code> are used	The field that the condition applies to. This value sets the <code>Condition.fieldId</code> property.

Parameter	Type	Required / Optional	Description
			Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser .
options.operator	string	required	The operator used by the condition. This value sets the Condition.operator parameter. Use the appropriate query.Operator enum value to pass in your argument. This enum holds all the supported values for this parameter.
options.values	string[] number[] boolean[] Date[] query.RelativeDate[] query.Period[]	required if options.fieldId and options.operator are used, and options.operator does not have a value of query.Operator.EMPTY or query.Operator.EMPTY_NOT	An array of values to use for the condition. This value sets the Condition.values property.
options.formula	string	required if options.fieldId is not used	The formula used to create the condition. This value sets the Condition.formula property. For more information on formulas, see the help topics Formulas in Search and SQL Expressions .
options.type	string	required if options.formula is used	If you use the options.formula parameter, use this parameter to explicitly define the formula's return type. This value sets the Condition.type property. Use the appropriate query.ReturnType enum value to pass in your argument. This enum holds all the supported values for this parameter.
options.aggregate	string	optional	Use this parameter to run an aggregate function on a condition. An aggregate function performs a calculation on the condition values and returns a single value. This value sets the Condition.aggregate property. Use the appropriate query.Aggregate enum value to pass in your argument. This enum holds all the supported values for this parameter.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myLocationJoin = mySalesRepJoin.autoJoin({
12   fieldId: 'location'
13 });
14
15 var firstCondition = myCustomerQuery.createCondition({
16   fieldId: 'id',
17   operator: query.Operator.EQUAL,
18   values: 107
19 });
20 var secondCondition = myCustomerQuery.createCondition({
21   fieldId: 'id',
22   operator: query.Operator.EQUAL,
23   values: 2647
24 });
25 var thirdCondition = mySalesRepJoin.createCondition({
26   fieldId: 'email',
27   operator: query.Operator.START_WITH_NOT,
28   values: 'foo'
29 });
30
31 myCustomerQuery.condition = myCustomerQuery.and(
32   thirdCondition, myCustomerQuery.not(
33     myCustomerQuery.or(firstCondition, secondCondition)
34   )
35 );
36
37 var resultSet = myCustomerQuery.run();
38 ...
39 // Add additional code

```

Query.createSort(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a sort based on the query.Query object. The query.Sort object describes a sort that is placed on a particular query result column.</p> <p>To create a sort:</p> <ul style="list-style-type: none"> ■ Use Search.createSort(options) to create a sort based on the initial query definition created with query.create(options). ■ Use Component.createSort(options) to create a sort based on a join relationship created with Query.autoJoin(options) or Component.autoJoin(options). ■ Assign all created sorts as array values to Query.sort. For an example, see Syntax.
---------------------------	---

	<p>Note: This method is a shortcut for the chained <code>Query.root</code> and <code>Component.createSort(options)</code>: <code>Query.root.createSort(options)</code>. The <code>Query.root</code> property references the root component, which is a <code>query.Component</code> object.</p>
Returns	<code>query.Sort</code>
Supported Script Types	<p>Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/query Module
Parent Object	<code>query.Query</code>
Sibling Object Members	Query Object Members
Since	2018.1

Parameters

<p>Note: The options parameter is a JavaScript object.</p>			
Parameter	Type	Required / Optional	Description
<code>options.column</code>	<code>query.Column</code>	required	The query result column that you want to sort by. This value sets the <code>Sort.column</code> property.
<code>optionsascending</code>	boolean	optional	<p>Whether the sort direction is ascending. This value sets the <code>Sort.ascending</code> property.</p> <p>The default value of this property is <code>true</code>, meaning that the sort direction is ascending. If you want the sort direction to be descending, set this property to <code>false</code>.</p>
<code>options.caseSensitive</code>	boolean	optional	<p>Whether the sort is case sensitive. This value sets the <code>Sort.caseSensitive</code> property.</p> <p>If a sort is case sensitive (and the sort direction is ascending), rows with column values that start with uppercase letters are listed before rows with column values that start with lowercase letters. If a sort is not case sensitive, uppercase and lowercase letters are treated the same. For example, the following list of items is sorted using a case-sensitive sort with a sort direction of ascending:</p> <ul style="list-style-type: none"> ■ Banana ■ Orange ■ apple ■ grapefruit ■ kiwi <p>Here is the same list of items sorted using a regular (not case-sensitive) sort with a sort direction of ascending:</p>

Parameter	Type	Required / Optional	Description
			<ul style="list-style-type: none"> ■ apple ■ Banana ■ grapefruit ■ kiwi ■ Orange <p>The default value of this property is false.</p>
options.locale	string	optional	<p>The locale to use for the sort. This value sets the Sort.locale property.</p> <p>A locale represents a combination of language and region, and it can affect how certain values (such as strings) are sorted. For example, languages that share the same alphabet may sort characters differently. Use this property to ensure that query results are sorted using locale-specific rules.</p> <p>Use the appropriate query.SortLocale enum value to pass in your argument. This enum holds all the supported values for this parameter.</p>
options.nullsLast	boolean	optional	<p>Whether query results with null values are listed at the end of the query results. This value sets the Sort.nullsLast property.</p> <p>The default value of this property is the value of the options.ascending property. For example, if the options.ascending property is set to true, the options.nullsLast property is also set to true.</p>

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   myCustomerQuery.createColumn({
16     fieldId: 'id'
17   }),
18   mySalesRepJoin.createColumn({
19     fieldId: 'entityid'
20   }),
21   mySalesRepJoin.createColumn({
22     fieldId: 'email'
23   }),
24   mySalesRepJoin.createColumn({

```

```

25     fieldId: 'hiredate'
26   })
27 ];
28
29 myCustomerQuery.sort = [
30   myCustomerQuery.createSort({
31     column: myCustomerQuery.columns[1]
32   }),
33   mySalesRepJoin.createSort({
34     column: mySalesRepJoin.columns[0],
35     ascending: false
36   })
37 ];
38
39 var resultSet = myCustomerQuery.run();
40 ...
41 // Add additional code

```

Query.join(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a join relationship.</p> <p>Important: This method is an alias to Query.autoJoin(options). Use Query.autoJoin(options) instead of this method to create simple joins. Use Query.joinFrom(options) and Query.joinTo(options) to create explicit directional joins.</p>
	<p>Use the method query.create(options) to create your initial query definition (query.Query). The initial query definition uses one query type. For available query types, see query.Type.</p> <p>After you create the initial query definition, use Query.join(options) to create your first join (query.Component). Then use Component.autoJoin(options) to create each subsequent join (query.Component).</p> <p>Note: This method is a shortcut for the chained Query.root and Component.join(options): Query.root.join(options). The Query.root property references the root component, which is a query.Component object.</p>
	<p>Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types.</p>
Returns	query.Component
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fieldId	string	required	<p>The column type (field type) that joins the parent component to the new component. This value determines the columns on which the components are joined and the type of the newly joined component.</p> <p>Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser.</p>

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEntityJoin = myTransactionQuery.join({
8   fieldId: 'entity'
9 });
10
11 myTransactionQuery.columns = [
12   myEntityJoin.createColumn({
13     fieldId: 'subsidiary'
14   })
15 ];
16
17 myTransactionQuery.sort = [
18   myTransactionQuery.createSort({
19     column: myTransactionQuery.columns[0],
20     ascending: false
21   })
22 ];
23
24 var results = myTransactionQuery.runPaged({
25   pageSize: 10
26 });
27 ...
28 // Add additional code

```

Query.joinFrom(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates an explicit directional join relationship from another component to this component (an inverse join). This method sets the Component.source property on the returned query.Component object.
---------------------------	--

	<p>Use the method <code>query.create(options)</code> to create your initial query definition (<code>query.Query</code>). The initial query definition uses one query type. For available query types, see <code>query.Type</code>.</p> <p>After you create the initial query definition, use this method to create your first join as an explicit directional join from another component to this component.</p>
	<p>Note: This method is a shortcut for the chained <code>Query.root</code> and <code>Component.joinFrom(options)</code>: <code>Query.root.joinFrom(options)</code>. The <code>Query.root</code> property references the root component, which is a <code>query.Component</code> object.</p>
	<p>Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types.</p>
Returns	<code>query.Component</code>
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	<code>query.Query</code>
Sibling Object Members	Query Object Members
Since	2018.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description
<code>options.fieldId</code>	string	required	<p>The column type (field type) that joins the parent component to the new component.</p> <p>Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser.</p>
<code>options.source</code>	string	required	<p>The query type of the component joined to this component. This value sets the <code>Component.source</code> property.</p> <p>This value can be described as the inverse relationship of this component, and it determines the source query type of the newly joined component.</p>

Errors

Error Code	Thrown If
<code>RELATIONSHIP_ALREADY_USED</code>	The specified join relationship already exists.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myEmployeeQuery = query.create({
4   type: query.Type.EMPLOYEE
5 });
6
7 var myTransactionJoin = myEmployeeQuery.joinFrom({
8   fieldId: 'entity',
9   source: 'transaction'
10 });
11
12 myEmployeeQuery.columns = [
13   myEmployeeQuery.createColumn({
14     fieldId: 'entityid'
15   }),
16   myTransactionJoin.createColumn({
17     fieldId: 'entity'
18   }),
19   myTransactionJoin.createColumn({
20     fieldId: 'daysoverduesearch'
21   })
22 ];
23 ...
24 // Add additional code

```

Query.joinTo(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates an explicit directional join relationship to another component from this component (a polymorphic join). This method sets the Component.target property on the returned query.Component object.</p> <p>Use the method query.create(options) to create your initial query definition (query.Query). The initial query definition uses one query type. For available query types, see query.Type.</p> <p>After you create the initial query definition, use this method to create your first join as an explicit directional join to another component from this component.</p>
	<p>Note: This method is a shortcut for the chained Query.root and Component.joinTo(options): <code>Query.root.autoJoin(options)</code>. The Query.root property references the root component, which is a query.Component object.</p>
	<p>Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types.</p>
Returns	query.Component
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fieldId	string	required	<p>The column type (field type) that joins the parent component to the new component.</p> <p>Obtain this value from the Analytics Browser. The Analytics Browser lists every record type and field that is available using SuiteAnalytics Workbook and the N/query module. For more information, see the help topic Analytics Browser.</p>
options.target	string	required	<p>The query type of the component joined to this component. This value sets the Component.target property.</p> <p>This value can be described as the polymorphic relationship of this component, and it determines the target query type of the newly joined component.</p>

Errors

Error Code	Thrown If
RELATIONSHIP_ALREADY_USED	The specified join relationship already exists.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEmployeeJoin = myTransactionQuery.joinTo({
8   fieldId: 'createdby',
9   target: 'employee'
10 });
11
12 myTransactionQuery.columns = [
13   myTransactionQuery.createColumn({
14     fieldId: 'entity'
15   }),

```

```
16 myEmployeeJoin.createColumn({
17   fieldId: 'entityid'
18 },
19 myEmployeeJoin.createColumn({
20   fieldId: 'email'
21 })
22 ];
23 ...
24 // Add additional code
```

Query.run()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Executes the query and returns the query result set. This method returns a maximum of 5000 results in the query result set. If a query matches more than 5000 results, you must use Query.runPaged() or Query.runPaged.promise() to retrieve the full set of results.
Returns	query.ResultSet
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```
1 // Add additional code
2 ...
3 var myCustomerQuery = query.createQuery({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   myCustomerQuery.createColumn({
16     fieldId: 'id'
17   }),
18   mySalesRepJoin.createColumn({
19     fieldId: 'entityid'
```

```

20     }),
21     mySalesRepJoin.createColumn({
22       fieldId: 'email'
23     }),
24     mySalesRepJoin.createColumn({
25       fieldId: 'hiredate'
26     })
27   ];
28
29 myCustomerQuery.sort = [
30   myCustomerQuery.createSort({
31     column: myCustomerQuery.columns[1]
32   }),
33   mySalesRepJoin.createSort({
34     column: mySalesRepJoin.columns[0],
35     ascending: false
36   })
37 ];
38
39 var resultSet = myCustomerQuery.run();
40 ...
41 // Add additional code

```

Query.run.promise()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Executes the query asynchronously and returns the query result set.  Note: The parameters and errors thrown for this method are the same as those for Query.run() . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Query.runPaged()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Executes the query and returns a query.PagedData object that represents the paged query results. You can iterate over this object to obtain each page of query results. For paged queries, the maximum number of result rows per page is 1000. The minimum number of result rows per page is 5, except for the last page in the result set (because the last page may include fewer than 5 results).
---------------------------	---

Returns	query.PagedData
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.pageSize	string	optional	The size of each page in the query results. The default page size is 50 results per page.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myEntityJoin = myTransactionQuery.autoJoin({
8   fieldId: 'entity'
9 });
10
11 myTransactionQuery.columns = [
12   myEntityJoin.createColumn({
13     name: 'subsidiary'
14   })
15 ];
16
17 myTransactionQuery.sort = [
18   myTransactionQuery.createSort({
19     column: myTransactionQuery.columns[0],
20     ascending: false
21   })
22 ];
23
24 var results = myTransactionQuery.runPaged({
25   pageSize: 10
26 });
27
28 // Use the count property to count the
29 // search results easily
30 var resultCount = myTransactionQuery.runPaged({
31   pageSize: 10
32 });

```

```

32 } ).count;
33 ...
34 // Add additional code

```

Query.runPaged.promise()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Executes the query asynchronously and returns a set of paged results.  Note: The parameters and errors thrown for this method are the same as those for Query.runPaged() . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Query.not()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new condition (a query.Condition object) that corresponds to a logical negation (NOT) of the argument passed to the method. The argument must be a query.Condition object. A condition narrows the query results. The query.Condition object acts in the same capacity as the search.Filter object in the N/search Module . The primary difference is that query.Condition objects can contain other query.Condition objects. To create conditions: <ul style="list-style-type: none">■ Use Query.createCondition(options) to create conditions for the initial query definition created with query.create(options).■ Use Component.createCondition(options) to create conditions for the join relationships created with Query.autoJoin(options) and Component.autoJoin(options).■ If you have multiple conditions, use them to create a new parent condition with the methods Query.and(), Query.or(), and Query.not().■ Assign your parent condition to Query.condition. For an example, see Syntax.
Returns	query.Condition
Supported Script Types	Client and server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Parameters

Parameter	Type	Required / Optional	Description
condition	query.Condition	required	One condition object.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myLocationJoin = mySalesRepJoin.autoJoin({
12   fieldId: 'location'
13 });
14
15 var firstCondition = myCustomerQuery.createCondition({
16   fieldId: 'id',
17   operator: query.Operator.EQUAL,
18   values: 107
19 });
20 var secondCondition = myCustomerQuery.createCondition({
21   fieldId: 'id',
22   operator: query.Operator.EQUAL,
23   values: 2647
24 });
25 var thirdCondition = mySalesRepJoin.createCondition({
26   fieldId: 'email',
27   operator: query.Operator.START_WITH_NOT,
28   values: 'foo'
29 });
30
31 myCustomerQuery.condition = myCustomerQuery.and(
32   thirdCondition, myCustomerQuery.not(
33     myCustomerQuery.or(firstCondition, secondCondition)
34   )
35 );
36
37 var resultSet = myCustomerQuery.run();
38 ...
39 // Add additional code

```

Query.or()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a new condition (a query.Condition object) that corresponds to a logical disjunction (OR) of the arguments passed to the method. The arguments must be one or more query.Condition objects.</p> <p>A condition narrows the query results. The query.Condition object acts in the same capacity as the search.Filter object in the N/search Module. The primary difference is that query.Condition objects can contain other query.Condition objects.</p> <p>To create conditions:</p> <ul style="list-style-type: none"> ■ Use Query.createCondition(options) to create conditions for the initial query definition created with query.create(options). ■ Use Component.createCondition(options) to create conditions for the join relationships created with Query.autoJoin(options) and Component.autoJoin(options). ■ If you have multiple conditions, use them to create a new parent condition with the methods Query.and(), Query.or(), and Query.not(). ■ Assign your parent condition to Query.condition. For an example, see Syntax.
Returns	query.Condition
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Parameters

Parameter	Type	Required / Optional	Description
condition1 — n	query.Condition[]	required	<p>One or more condition objects.</p> <p>There is no limit on the number of conditions you can specify.</p>

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER

```

```

5  });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var myLocationJoin = mySalesRepJoin.autoJoin({
12   fieldId: 'location'
13 });
14
15 var firstCondition = myCustomerQuery.createCondition({
16   fieldId: 'id',
17   operator: query.Operator.EQUAL,
18   values: 107
19 });
20 var secondCondition = myCustomerQuery.createCondition({
21   fieldId: 'id',
22   operator: query.Operator.EQUAL,
23   values: 2647
24 });
25 var thirdCondition = mySalesRepJoin.createCondition({
26   fieldId: 'email',
27   operator: query.Operator.START_WITH_NOT,
28   values: 'foo'});
29
30 myCustomerQuery.condition = myCustomerQuery.and(
31   thirdCondition, myCustomerQuery.not(
32     myCustomerQuery.or(firstCondition, secondCondition)
33   )
34 );
35
36 var resultSet = myCustomerQuery.run();
37 ...
38 // Add additional code

```

Query.toSuiteQL()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Converts this query.Query object to its corresponding SuiteQL representation.</p>
	<p>This method returns a query.SuiteQL object that represents the same query as the original query.Query object. This object includes the SuiteQL.columns, SuiteQL.params, SuiteQL.query, and SuiteQL.type properties. You can run the query using SuiteQL.run(), or you can run the query as a paged query using SuiteQL.runPaged(options).</p>
	<div style="border: 1px solid #f0e68c; padding: 10px;"> <p>Important: The resulting SuiteQL query string (contained in the SuiteQL.query property) does not include any aliases you set on query result columns in the original query.Query object. For more information about aliases, see Column.alias.</p> </div>
	<p>For more information and examples of using SuiteQL in the N/query module, see SuiteQL in the N/query Module. For more information about SuiteQL in general, see the help topic SuiteQL.</p>
Returns	<p>query.SuiteQL</p>
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	<p>None</p>
Module	<p>N/query Module</p>
Parent Object	<p>query.Query</p>

Sibling Object Members	Query Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // myQuery is an existing query.Query object
4 var mySuiteQLQuery = myQuery.toSuiteQL();
5
6 var results = mySuiteQLQuery.run();
7 ...
8 // Add additional code

```

Query.child



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A reference to children of this component. The value of this property is an object of key/value pairs. Each key is the name of a child component. Each respective value is the corresponding query.Component object. The object values are set with the execution of Query.autoJoin(options) and Component.autoJoin(options) . The order of the key/value pairs reflects the parent/child hierarchy.
Type	Object
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });

```

```

10 // Add additional code
11 ...
12 var myTaskJoin = myCustomerQuery.autoJoin({
13   fieldId: 'task'
14 });
15 var theChild = myCustomerQuery.child;
16 ...
17 //Add additional code

```

Query.columns



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of result columns (query.Column objects) returned from the query. The query.Column object is the equivalent of the search.Column object in the N/search Module . The query.Column object describes a field type (column) that is returned from the query results. To create columns: <ul style="list-style-type: none">■ Use Query.createColumn(options) to create conditions on the initial query definition created with query.create(options).■ Use Component.createColumn(options) to create conditions on the join relationships created with Query.autoJoin(options) and Component.autoJoin(options).■ Assign all created columns as array values to <code>Query.columns</code>. For an example, see Syntax.
Type	query.Column[]
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   mySalesRepJoin.createColumn({
16     fieldId: 'firstname'
17   }),

```

```

18     mySalesRepJoin.createColumn({
19         fieldId: 'email'
20     })
21 ];
22 ...
23 // Add additional code

```

Query.condition



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The simple or nested condition (a query.Condition object) that narrows the query results. The query.Condition object acts in the same capacity as the search.Filter object in the N/search Module . The primary difference is that query.Condition objects can contain other query.Condition objects. To create conditions: <ul style="list-style-type: none"> ■ Use Query.createCondition(options) to create conditions for the initial query definition created with query.create(options). ■ Use Component.createCondition(options) to create conditions for the join relationships created with Query.autoJoin(options) and Component.autoJoin(options). ■ If you have multiple conditions, use them to create a new nested condition with the methods Query.and(), Query.or(), and Query.not(). ■ Assign your simple or nested condition to <code>Query.condition</code>. For an example, see Syntax.
Type	query.Condition
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4     type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8     fieldId: 'salesrep'
9 });
10
11 var myFirstCondition = myCustomerQuery.createCondition({
12     fieldId: 'id',
13     operator: query.Operator.EQUAL,
14     values: 107
15 });
16
17 var mySecondCondition = myCustomerQuery.createCondition({

```

```

18     fieldId: 'id',
19     operator: query.Operator.EQUAL,
20     values: 2647
21   });
22
23 var myThirdCondition = myCustomerQuery.createCondition({
24   fieldId: 'email',
25   operator: query.Operator.START_WITH_NOT,
26   values: 'foo'
27 });
28
29 myCustomerQuery.condition = myCustomerQuery.and(
30   myThirdCondition, myCustomerQuery.not(
31     myCustomerQuery.or(myFirstCondition, mySecondCondition)
32   )
33 );
34 ...
35 // Add additional code

```

Query.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the query definition. This property has a value only for existing queries that are loaded using query.load(options) . If you create a query using query.create(options) but do not save it, this property is null.
Type	number (read-only)
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myLoadedQuery = query.load({
4   id: 'custworkbook237'
5 });
6
7 var theId = myLoadedQuery.id;
8 ...
9 // Add additional code

```

Query.name

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of the query definition. This property has a value only for existing queries that are loaded using <code>query.load(options)</code> . If you create a query using <code>query.create(options)</code> but do not save it, this property is null.
Type	string (read-only)
Module	N/query Module
Parent Object	<code>query.Query</code>
Sibling Object Members	Query Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myLoadedQuery = query.load({
4   id: 'custworkbook237'
5 });
6
7 var theName = myLoadedQuery.name;
8 ...
9 // Add additional code

```

Query.root

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The root component of the query definition. The initial <code>query.Component</code> object is called the root component. It encapsulates the initial query type passed to <code>query.create(options)</code> . The root component is automatically created with the <code>query.Query</code> object and is a child of the <code>query.Query</code> object.
Type	<code>query.Component</code> (read-only)
Module	N/query Module
Parent Object	<code>query.Query</code>

Sibling Object Members	Query Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var theRoot = myCustomerQuery.root;
8 ...
9 // Add additional code

```

Query.sort

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of query.Sort objects used for sorting. This object encapsulates a sort based on the query.Query or query.Component object. The query.Sort object describes a sort that is placed on a particular query result column. To create a sort: <ul style="list-style-type: none"> ■ Use Query.createSort(options) to create a sort based on the initial query definition created with query.create(options). ■ Use Component.createSort(options) to create a sort based on a join relationship created with Query.autoJoin(options) or Component.autoJoin(options). ■ Assign all created sorts as array values to Query.sort. For an example, see Syntax.
Type	query.Sort[]
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({

```

```

4     type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8     fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12     myCustomerQuery.createColumn({
13         fieldId: 'entityid'
14     }),
15     mySalesRepJoin.createColumn({
16         fieldId: 'firstname'
17     }),
18     mySalesRepJoin.createColumn({
19         fieldId: 'email'
20     })
21 ];
22
23 myCustomerQuery.sort = [
24     myCustomerQuery.createSort({
25         column: myCustomerQuery.columns[1]
26     }),
27     mySalesRepJoin.createSort({
28         column: myCustomerQuery.columns[0],
29         ascending: false
30     })
31 ];
32 ...
33 // Add additional code

```

Query.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The initial query type of the query definition. This property is set during the execution of query.create(options) .
Type	string (read-only)
Module	N/query Module
Parent Object	query.Query
Sibling Object Members	Query Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4     type: query.Type.CUSTOMER
5 });
6
7 var theType = myCustomerQuery.type;
8 ...
9 // Add additional code

```

query.RelativeDate

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	A relative date to use in query conditions. Use query.createRelativeDate(options) to create this object. After you create this object, you can use it in the values parameter of Query.createCondition(options) or Component.createCondition(options) . This object represents a specific moment in time, and you can use it to create query conditions using operators from the query.Operator enum, such as <code>query.Operator.AFTER</code> , <code>query.Operator.BEFORE</code> , and <code>query.Operator.WITHIN</code> . For more information about relative dates, see Relative Dates in the N/query Module .
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	RelativeDate Object Members
Since	2019.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var myEndDate = query.createRelativeDate({
4   dateId: query.DateId.WEEKS_AGO,
5   value: 2
6 });
7
8 var myComplexCondition = myQuery.createCondition({
9   fieldId: 'trandate',
10  operator: query.Operator.WITHIN,
11  values: [query.RelativeDateRange.THREE_FISCAL_YEARS_AGO.start, myEndDate]
12 });
13 ...
14 //Add additional code

```

RelativeDate.dateId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The date ID of the relative date. For relative dates that you create using query.createRelativeDate(options) , the value of this property is set when that method is executed. For relative dates that are included in the query.RelativeDateRange enum, the value of this property is always available (for example, <code>query.RelativeDateRange.YESTERDAY.dateId</code>). This property uses values from the query.DateId enum.
Type	string (read-only)

Module	N/query Module
Parent Object	query.RelativeDate
Sibling Object Members	RelativeDate Object Members
Since	2019.1

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myRelativeDate = query.createRelativeDate({
4     dateId: query.DateId.WEEKS_AGO,
5     value: 2
6 });
7
8 var theDateId = myRelativeDate.dateId;
9 ...
10 // Add additional code

```

RelativeDate.end

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The end of the relative date. For relative dates that you create using query.createRelativeDate(options) , the value of this property is set when that method is executed. For relative date ranges that are included in the query.RelativeDateRange enum, the value of this property is always available (for example, <code>query.RelativeDateRange.YESTERDAY.end</code>).
Type	Object (read-only)
Module	N/query Module
Parent Object	query.RelativeDate
Sibling Object Members	RelativeDate Object Members
Since	2019.1

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myRelativeDate = query.createRelativeDate({
4     dateId: query.DateId.WEEKS_AGO,
5     value: 2
6 });

```

```

7
8 var theEnd = myRelativeDate.end;
9 ...
10 // Add additional code

```

RelativeDate.interval



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The interval that the relative date represents. For relative dates that you create using query.createRelativeDate(options) , the value of this property is set when that method is executed. For relative date ranges that are included in the query.RelativeDateRange enum, the value of this property is always available (for example, <code>query.RelativeDateRange.YESTERDAY.interval</code>).
Type	Object (read-only)
Module	N/query Module
Parent Object	query.RelativeDate
Sibling Object Members	RelativeDate Object Members
Since	2019.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myRelativeDate = query.createRelativeDate({
4     dateId: query.DateId.WEEKS_AGO,
5     value: 2
6 });
7
8 var theInterval = myRelativeDate.interval;
9 ...
10 // Add additional code

```

RelativeDate.isRange



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether the relative date represents a range of dates or a specific moment in time. For relative date ranges that you obtain from the query.RelativeDateRange enum, the value of this property is true (the relative date represents a range of dates). For all other relative dates (such as those that you create using query.createRelativeDate(options)), the value of this property is false (the relative date represents a specific moment in time).
-----------------------------	---

Type	boolean (read-only)
Module	N/query Module
Parent Object	query.RelativeDate
Sibling Object Members	RelativeDate Object Members
Since	2019.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myRelativeDate = query.createRelativeDate({
4   dateId: query.DateId.WEEKS_AGO,
5   value: 2
6 });
7
8 // isARange is false
9 var isARange = myRelativeDate.isRange;
10
11 // isAnotherRange is true
12 var isAnotherRange = query.RelativeDateRange.LAST_MONTH_ONE_FISCAL_YEAR_AGO.isRange;
13 ...
14 // Add additional code

```

RelativeDate.start

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The start of the relative date. For relative dates that you create using query.createRelativeDate(options) , the value of this property is set when that method is executed. For relative date ranges that are included in the query.RelativeDateRange enum, the value of this property is always available (for example, <code>query.RelativeDateRange.YESTERDAY.start</code>).
Type	Object (read-only)
Module	N/query Module
Parent Object	query.RelativeDate
Sibling Object Members	RelativeDate Object Members
Since	2019.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 var myRelativeDate = query.createRelativeDate({
4   dateId: query.DateId.WEEKS_AGO,
5   value: 2
6 });
7
8 var theStart = myRelativeDate.start;
9 ...
10 // Add additional code

```

RelativeDate.value



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The value of the relative date range. For relative dates that you create using query.createRelativeDate(options) , the value of this property is set when that method is executed. For relative date ranges that are included in the query.RelativeDateRange enum, the value of this property is undefined (for example, <code>query.RelativeDateRange.YESTERDAY.value</code> is undefined).
Type	number (read-only)
Module	N/query Module
Parent Object	query.RelativeDate
Sibling Object Members	RelativeDate Object Members
Since	2019.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myRelativeDate = query.createRelativeDate({
4   dateId: query.DateId.WEEKS_AGO,
5   value: 2
6 });
7
8 var theValue = myRelativeDate.value;
9 ...
10 // Add additional code

```

query.Result



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A single row of the result set (query.ResultSet).
Supported Script Types	Client and server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	Result Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   }),
11   myCustomerQuery.createColumn({
12     fieldId: 'firstname'
13   }),
14   myCustomerQuery.createColumn({
15     fieldId: 'email'
16   })
17 ];
18
19 var queryResultSet = myCustomerQuery.run();
20
21 // Fetch results using an iterator
22 var iterator = queryResultSet.iterator();
23 iterator.each(function(result) {
24   var currentResult = result.value;
25   log.debug(currentResult);
26   return true;
27 });
28
29 // Alternatively, fetch results using a loop
30 var queryResults = queryResultSet.results;
31 for (var i = 0; i < queryResults.length; i++) {
32   var currentResult = queryResults[i];
33   log.debug(currentResult);
34 }
35 ...
36 // Add additional code

```

Result.asMap()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the query result as a mapped result. A mapped result is a JavaScript object with key-value pairs. In this object, the key is either the field ID or the alias that was used for the corresponding query.Column object.
Returns	Object
Supported Script Types	Client and server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.Result
Sibling Object Members	Result Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid',
14     alias: 'cust'
15   }),
16   myCustomerQuery.createColumn({
17     fieldId: 'id'
18   }),
19   mySalesRepJoin.createColumn({
20     fieldId: 'entityid'
21   }),
22   mySalesRepJoin.createColumn({
23     fieldId: 'email'
24   }),
25   mySalesRepJoin.createColumn({
26     fieldId: 'hiredate'
27   })
28 ];
29
30 var resultSet = myCustomerQuery.run();
31
32 for (var i = 0; i < resultSet.results.length; i++) {
33   var mResult = resultSet.results[i].asMap();
34   log.debug(mResult);
35 }
36 ...
37 // Add additional code

```

Result.values



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The result values. Value types correspond to the ResultSet.types property. Array values correspond to the array values for ResultSet.columns .
-----------------------------	--

Type	Array<string number boolean null> (read-only)
Module	N/query Module
Parent Object	query.Result
Sibling Object Members	Result Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10  }),
11  myCustomerQuery.createColumn({
12    fieldId: 'email'
13  })
14];
15
16 var queryResultSet = myCustomerQuery.run();
17
18 var queryResults = queryResultSet.results;
19 var myFirstResult = queryResults[0];
20 var theValues = myFirstResult.values;
21 ...
22 // Add additional code

```

query.ResultSet



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The set of results returned by the query. Use Query.run() or Query.run.promise() to create this object. The maximum number of results in a ResultSet object is 5000. If a query matches more than 5000 results, you must use Query.runPaged() or Query.runPaged.promise() to retrieve the full set of results.
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module
Methods and Properties	ResultSet Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10}),
11  myCustomerQuery.createColumn({
12    fieldId: 'email'
13 })
14];
15
16 var resultSet = myCustomerQuery.run();
17
18 var results = resultSet.results;
19 for (var i = results.length - 1; i >= 0; i--)
20   log.debug(results[i].values);
21 ...
22 // Add additional code

```

ResultSet.asMappedResults()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the query result set as an array of mapped results. A mapped result is a JavaScript object with key-value pairs. In this object, the key is either the field ID or the alias that was used for the corresponding query.Column object. When you call this method, Result.asList () is called on each query.Result object in the result set.
Returns	Object[]
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.ResultSet
Sibling Object Members	ResultSet Object Members
Since	2019.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid',
14     alias: 'cust'
15   }),
16   myCustomerQuery.createColumn({
17     fieldId: 'id'
18   }),
19   mySalesRepJoin.createColumn({
20     fieldId: 'entityid'
21   }),
22   mySalesRepJoin.createColumn({
23     fieldId: 'email'
24   }),
25   mySalesRepJoin.createColumn({
26     fieldId: 'hiredate'
27   })
28 ];
29
30 var resultSet = myCustomerQuery.run();
31 var mrSet = resultSet.asMappedResults();
32 ...
33 // Add additional code

```

ResultSet.iterator()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Standard SuiteScript 2.0 object for iterating through results
Returns	Iterator object
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Parent Object	query.ResultSet
Sibling Object Members	ResultSet Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER

```

```

5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10}),
11 myCustomerQuery.createColumn({
12   fieldId: 'firstname'
13}),
14 myCustomerQuery.createColumn({
15   fieldId: 'email'
16 })
17];
18
19 var queryResultSet = myCustomerQuery.run();
20
21 // Fetch results using an iterator
22 var iterator = queryResultSet.iterator();
23 iterator.each(function(result) {
24   var currentResult = result.value;
25   log.debug(currentResult);
26   return true;
27 });
28
29 // Alternatively, fetch results using a loop
30 var queryResults = queryResultSet.results;
31 for (var i = 0; i < queryResults.length; i++) {
32   var currentResult = queryResults[i];
33   log.debug(currentResult);
34 }
35 ...
36 // Add additional code

```

ResultSet.columns



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of query return column references. The <code>ResultSet.columns</code> array values correspond with the ResultSet.types array values.
Type	<code>query.Column[]</code> (read-only)
Module	N/query Module
Parent Object	query.ResultSet
Sibling Object Members	ResultSet Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'

```

```

10  });
11  myCustomerQuery.createColumn({
12    fieldId: 'email'
13  })
14];
15
16 var queryResultSet = myCustomerQuery.run();
17
18 var theColumns = queryResultSet.columns;
19 ...
20 // Add additional code

```

ResultSet.results

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of query.Result objects.
Type	query.Result[] (read-only)
Module	N/query Module
Parent Object	query.ResultSet
Sibling Object Members	ResultSet Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10  }),
11  myCustomerQuery.createColumn({
12    fieldId: 'email'
13  })
14];
15
16 var queryResultSet = myCustomerQuery.run();
17
18 var theResults = queryResultSet.results;
19 ...
20 // Add additional code

```

ResultSet.types

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of the return types for ResultSet.results .
-----------------------------	--

	The ResultSet.types array values correspond with the ResultSet.columns array values.
Type	string[] (read-only)
Module	N/query Module
Parent Object	query.ResultSet
Sibling Object Members	ResultSet Object Members
Since	2018.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   }),
11   myCustomerQuery.createColumn({
12     fieldId: 'email'
13   })
14 ];
15
16 var queryResultSet = myCustomerQuery.run();
17
18 var theTypes = queryResultSet.types;
19 ...
20 // Add additional code

```

query.Sort



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A sort based on the query.Query or query.Component object. The <code>query.Sort</code> object describes a sort that is placed on a particular query result column. To create a sort: <ul style="list-style-type: none">■ Use Query.createSort(options) to create a sort based on the initial query definition created with query.create(options).■ Use Component.createSort(options) to create a sort based on a join relationship created with Query.autoJoin(options) or Component.autoJoin(options).■ Assign all created sorts as array values to Query.sort. For an example, see Syntax.
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/query Module

Methods and Properties	Sort Object Members
Since	2018.1

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   myCustomerQuery.createColumn({
16     fieldId: 'id'
17   }),
18   mySalesRepJoin.createColumn({
19     fieldId: 'entityid'
20   }),
21   mySalesRepJoin.createColumn({
22     fieldId: 'email'
23   }),
24   mySalesRepJoin.createColumn({
25     fieldId: 'hiredate'
26   })
27 ];
28
29 myCustomerQuery.sort = [
30   myCustomerQuery.createSort({
31     column: myCustomerQuery.columns[1]
32   }),
33   mySalesRepJoin.createSort({
34     column: mySalesRepJoin.columns[0],
35     ascending: false
36   })
37 ];
38
39 var resultSet = myCustomerQuery.run();
40 ...
41 // Add additional code

```

Sort.ascending

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether the sort direction is ascending. This property is set during the execution of Query.createSort(options) and Component.createSort(options) . The default value of this property is true, meaning that the sort direction is ascending. If you want the sort direction to be descending, set this property to false.
-----------------------------	--

Type	boolean
Module	N/query Module
Parent Object	query.Sort
Sibling Object Members	Sort Object Members
Since	2018.2

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 myCustomerQuery.sort = [
14   myCustomerQuery.createSort({
15     column: myCustomerQuery.columns[0],
16     ascending: false,
17     caseSensitive: true,
18     locale: query.SortLocale.EN_CA,
19     nullsLast: false
20   })
21 ];
22
23 var theAscending = myCustomerQuery.sort[0].ascending;
24 ...
25 // Add additional code

```

Sort.caseSensitive

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>Whether the sort is case sensitive.</p> <p>This property is set during the execution of Query.createSort(options) and Component.createSort(options).</p> <p>If a sort is case sensitive (and the sort direction is ascending), rows with column values that start with uppercase letters are listed before rows with column values that start with lowercase letters. If a sort is not case sensitive, uppercase and lowercase letters are treated the same. For example, the following list of items is sorted using a case-sensitive sort with a sort direction of ascending:</p> <ul style="list-style-type: none"> ■ Banana ■ Orange ■ apple ■ grapefruit
-----------------------------	--

	<ul style="list-style-type: none"> ■ kiwi <p>Here is the same list of items sorted using a regular (not case-sensitive) sort with a sort direction of ascending:</p> <ul style="list-style-type: none"> ■ apple ■ Banana ■ grapefruit ■ kiwi ■ Orange <p>The default value of this property is false.</p>
Type	boolean
Module	N/query Module
Parent Object	query.Sort
Sibling Object Members	Sort Object Members
Since	2018.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4     type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8     myCustomerQuery.createColumn({
9         fieldId: 'entityid'
10    })
11 ];
12
13 myCustomerQuery.sort = [
14     myCustomerQuery.createSort({
15         column: myCustomerQuery.columns[0],
16         ascending: false,
17         caseSensitive: true,
18         locale: query.SortLocale.EN_CA,
19         nullsLast: false
20    })
21 ];
22
23 var theCaseSensitive = myCustomerQuery.sort[0].caseSensitive;
24 ...
25 // Add additional code

```

Sort.column



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The query result column that the query results are sorted by.
-----------------------------	---

	This property is set during the execution of Query.createSort(options) and Component.createSort(options) .
Type	<code>query.Column</code> (read-only)
Module	N/query Module
Parent Object	<code>query.Sort</code>
Sibling Object Members	Sort Object Members
Since	2018.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 myCustomerQuery.sort = [
14   myCustomerQuery.createSort({
15     column: myCustomerQuery.columns[0],
16     ascending: false,
17     caseSensitive: true,
18     locale: query.SortLocale.EN_CA,
19     nullsLast: false
20   })
21 ];
22
23 var theColumn = myCustomerQuery.sort[0].column;
24 ...
25 // Add additional code

```

Sort.locale



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The locale to use for the sort.</p> <p>This property uses values from the <code>query.SortLocale</code> enum. This property is set during the execution of Query.createSort(options) and Component.createSort(options).</p> <p>A locale represents a combination of language and region, and it can affect how certain values (such as strings) are sorted. For example, languages that share the same alphabet may sort characters differently. Use this property to ensure that query results are sorted using locale-specific rules.</p>
Type	<code>string</code>

Module	N/query Module
Parent Object	query.Sort
Sibling Object Members	Sort Object Members
Since	2018.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 myCustomerQuery.sort = [
14   myCustomerQuery.createSort({
15     column: myCustomerQuery.columns[0],
16     ascending: false,
17     caseSensitive: true,
18     locale: query.SortLocale.EN_CA,
19     nullsLast: false
20   })
21 ];
22
23 var theLocale = myCustomerQuery.sort[0].locale;
24 ...
25 // Add additional code

```

Sort.nullsLast



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether query results with null values are listed at the end of the query results. This property is set during the execution of Query.createSort(options) and Component.createSort(options) . The default value of this property is the value of the Sort.ascending property. For example, if the Sort.ascending property is set to true, the Sort.nullsLast property is also set to true.
Type	boolean
Module	N/query Module
Parent Object	query.Sort
Sibling Object Members	Sort Object Members

Since	2018.2
-------	--------

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4     type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8     myCustomerQuery.createColumn({
9         fieldId: 'entityid'
10    })
11];
12
13 myCustomerQuery.sort = [
14     myCustomerQuery.createSort({
15         column: myCustomerQuery.columns[0],
16         ascending: false,
17         caseSensitive: true,
18         locale: query.SortLocale.EN_CA,
19         nullsLast: false
20    })
21];
22
23 var theNullsLast = myCustomerQuery.sort[0].nullsLast;
24 ...
25 // Add additional code

```

query.SuiteQL

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>A SuiteQL query.</p> <p>SuiteQL is a query language based on the SQL-92 revision of the SQL database query language. It provides advanced query capabilities you can use to access your NetSuite records and data.</p> <p>Use Query.toSuiteQL() to create this object. This method converts an existing query.Query object to its corresponding SuiteQL representation as a query.SuiteQL object. You can use SuiteQL.run() to run the query and obtain the results as a query.ResultSet object. You can also use SuiteQL.runPaged(options) to run the query as a paged query and obtain the results as a query.PagedData object.</p> <p>When you convert a query.Query object to a query.SuiteQL object, the resulting SuiteQL query is the same as the original query. It includes the same query result columns, sort order, and conditions that were set on the original query. When you run the resulting SuiteQL query using SuiteQL.run() or SuiteQL.runPaged(options), you receive the same results as you would if you ran the original query using Query.run() or Query.runPaged().</p> <p>For more information and examples of using SuiteQL in the N/query module, see SuiteQL in the N/query Module. For more information about SuiteQL in general, see the help topic SuiteQL.</p>
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>

Module	N/query Module
Methods and Properties	SuiteQL Object Members
Since	2020.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // myQuery is an existing query.Query object
4 var mySuiteQLQuery = myQuery.toSuiteQL();
5
6 var results = mySuiteQLQuery.run();
7 ...
8 // Add additional code

```

SuiteQL.columns

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The result columns to be returned from the query.</p> <p>This property is an array of <code>query.Column</code> objects. When you use <code>Query.toSuiteQL()</code> to convert an existing <code>query.Query</code> object to SuiteQL, this property contains the same <code>query.Column</code> objects that were specified for the original query.</p> <p> Important: The SuiteQL query string in a <code>query.SuiteQL</code> object (contained in the <code>SuiteQL.query</code> property) does not include any aliases you set on query result columns in the original <code>query.Query</code> object. For more information about aliases, see Column.alias.</p>
Type	<code>query.Column[]</code> (read-only)
Module	N/query Module
Parent Object	<code>query.SuiteQL</code>
Sibling Object Members	SuiteQL Object Members
Since	2020.1

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 //myQuery is an existing query.Query object
4 var mySuiteQLQuery = myQuery.toSuiteQL();
5
6 var theColumns = mySuiteQLQuery.columns;
7 ...
8 // Add additional code

```

SuiteQL.params



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The parameters for the query.</p> <p>In SuiteQL, query conditions are represented using the WHERE clause and a set of parameters. In a SuiteQL string, parameter values for conditions are represented using question marks (?), and the params property includes a list of the parameter values to use when the query runs. If the query uses more than one parameter, the order of the values in the params property matches the order the parameters appear in the query string.</p> <p>For example, consider the following query.Condition object in a query for customer records:</p> <pre> 1 myQueryObject.createCondition({ 2 fieldId: 'creditlimit', 3 operator: query.Operator.LESS_OR_EQUAL, 4 values: [50000] 5 }); </pre> <p>If you use Query.toSuiteQL() to convert this query to SuiteQL, the resulting SuiteQL query string includes the following WHERE clause:</p> <pre> 1 WHERE customer.creditlimit <= ? </pre> <p>In the resulting query.SuiteQL object, the params property includes the value 50000.</p>
Type	Array<string number boolean> (read-only)
Module	N/query Module
Parent Object	query.SuiteQL
Sibling Object Members	SuiteQL Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 //myQuery is an existing query.Query object
4 var mySuiteQLQuery = myQuery.toSuiteQL();
5
6 var theParams = mySuiteQLQuery.params;
7 ...

```

```
8 // Add additional code
```

SuiteQL.query



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The string representation of the SuiteQL query.</p> <p>This string can contain the following types of elements:</p> <ul style="list-style-type: none"> ■ SQL clauses, such as SELECT, FROM, and WHERE ■ Record or table names, such as customer ■ Field names using dot notation, such as customer.entityid ■ Operators for conditions, such as = and >= ■ Formatting characters, such as newline characters (\n) <p>This string can also contain field formatting metadata, such as /*{entityid#RAW}*/. When you use Query.toSuiteQL() to convert a constructed query to its SuiteQL query equivalent, metadata may be added to the query string to indicate the formatting (or context) of fields in the query. For example, /*{entityid#RAW}*/ metadata indicates that the entityid field is formatted using the query.FieldContext.RAW field context from the query.FieldContext enum. This metadata is added as comments (using /* and */) and does not affect query execution or the query results.</p>
Type	string (read-only)
Module	N/query Module
Parent Object	query.SuiteQL
Sibling Object Members	SuiteQL Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```
1 // Add additional code
2 ...
3 //myQuery is an existing query.Query object
4 var mySuiteQLQuery = myQuery.toSuiteQL();
5
6 var theQuery = mySuiteQLQuery.query;
7 ...
8 //Add additional code
```

SuiteQL.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The type of the query.
-----------------------------	------------------------

	This property uses values from the query.Type enum. When you use Query.toSuiteQL() to convert an existing query.Query object to SuiteQL, this property contains the same type that was specified for the original query.
Type	string (read-only)
Module	N/query Module
Parent Object	query.SuiteQL
Sibling Object Members	SuiteQL Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // myQuery is an existing query.Query object
4 var mySuiteQLQuery = myQuery.toSuiteQL();
5
6 var theType = mySuiteQLQuery.type;
7 ...
8 // Add additional code

```

SuiteQL.run()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Runs the SuiteQL query and returns the query results.</p> <p>You can use this method to run the SuiteQL query and obtain the results as a query.ResultSet object. If you want to run the SuiteQL query as a paged query, use SuiteQL.runPaged(options).</p> <p>Calling this method is equivalent to calling query.runSuiteQL(options) and passing the query.SuiteQL object as a parameter:</p> <pre> 1 // These two query calls return the same result set 2 var firstResultSet = mySuiteQLObject.run(); 3 var secondResultSet = query.runSuiteQL(mySuiteQLObject); </pre> <p>Note: If the SuiteAnalytics Connect feature is enabled in your NetSuite account, there is no limit to the number of results this method can return. If the SuiteAnalytics Connect feature is not enabled, this method can return a maximum of 100,000 results. For more information about SuiteAnalytics Connect, see the help topic SuiteAnalytics Connect.</p> <p>For more information and examples of using SuiteQL in the N/query module, see SuiteQL in the N/query Module. For more information about SuiteQL in general, see the help topic SuiteQL.</p>
Returns	query.ResultSet

Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/query Module
Parent Object	query.SuiteQL
Sibling Object Members	SuiteQL Object Members
Since	2020.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // myQuery is an existing query.Query object
4 var mySuiteQLQuery = myQuery.toSuiteQL();
5
6 var results = mySuiteQLQuery.run();
7 ...
8 // Add additional code

```

SuiteQL.runPaged(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Runs the SuiteQL query as a paged query and returns the paged query results.</p> <p>You can use this method to run the SuiteQL query and obtain the results as a query.PagedData object. If you want to run the SuiteQL query as a non-paged query, use SuiteQL.run().</p> <p>Note: If the SuiteAnalytics Connect feature is enabled in your NetSuite account, there is no limit to the number of results this method can return. If the SuiteAnalytics Connect feature is not enabled, this method can return a maximum of 100,000 results across all pages in the result set. For more information about SuiteAnalytics Connect, see the help topic SuiteAnalytics Connect.</p> <p>For more information and examples of using SuiteQL in the N/query module, see SuiteQL in the N/query Module. For more information about SuiteQL in general, see the help topic SuiteQL.</p>
Returns	query.PagedData
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units

Module	N/query Module
Parent Object	query.SuiteQL
Sibling Object Members	SuiteQL Object Members
Since	2020.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.pageSize	number	optional	The size of each page in the query results. The default value is 50 results per page. The minimum page size is 5 results per page, and the maximum page size is 1000 results per page.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // myQuery is an existing query.Query object
4 var mySuiteQLQuery = myQuery.toSuiteQL();
5
6 var results = mySuiteQLQuery.runPaged({
7   pageSize: 20
8 });
9 ...
10 // Add additional code

```

query.create(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a query.Query object.</p> <p>Use this method to create your initial query definition. The initial query definition uses one query type. For available query types, see query.Type.</p> <p>After you create the initial query definition, use Query.autoJoin(options) to create your first join. Then use Query.autoJoin(options) or Component.autoJoin(options) to create all subsequent joins.</p> <p>For standard record types, the query type that you specify is validated immediately and must be one of the values in the query.Type enum. For custom record types, the query type that you specify is not validated until the query is executed using Query.run() or Query.runPaged() (or</p>
---------------------------	---

	<p>using the promise versions of these methods). If you specify a query type for a custom record type that does not exist, this method allows you to create the query and does not throw an error. However, when you execute the query, an error is thrown.</p> <p>Important: The N/query module lets you create and run queries using the SuiteAnalytics Workbook query engine. You can use the N/query module to load and delete existing queries, but you cannot save queries. You can save queries using the SuiteAnalytics Workbook interface. For more information, see the help topic Navigating SuiteAnalytics Workbook.</p> <p>For more information about creating queries, see Scripting with the N/query Module.</p>
Returns	query.Query
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.1

Parameters

<p>Note: The options parameter is a JavaScript object.</p>			
Parameter	Type	Required / Optional	Description
options.type	string	required	<p>The query type that you want to use for the initial query definition.</p> <p>Use the query.Type enum to set this value (for an example, see the help topic Syntax).</p> <p>When you execute <code>query.create(options)</code>, the Query.type property is set based on this value.</p> <p>Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types.</p>
options.columns	Object[]	optional	<p>An array of objects to be used as query columns.</p> <p>If you specify a value for this parameter, the Query.createColumn(options) method is called on each object in the array.</p>
options.condition	Object	optional	<p>A condition for the query.</p> <p>If you specify a value for this parameter, the Query.createCondition(options) method is called on the object you specify.</p>

Parameter	Type	Required / Optional	Description
options.sort	Object[]	optional	An array of objects representing sort options. If you specify a value for this parameter, the Query.createColumn(options) and Query.createSort(options) methods are called on each object in the array.

Errors

Error Code	Thrown If
INVALID_RCRD_TYPE	The specified query type is invalid. i Note: This error is not thrown if you specify a custom record type as the query type. Custom record types are validated when the query is executed using Query.run() or Query.runPaged() (or using the promise versions of these methods).

Syntax

! **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 var mySalesRepJoin = myCustomerQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 myCustomerQuery.columns = [
12   myCustomerQuery.createColumn({
13     fieldId: 'entityid'
14   }),
15   myCustomerQuery.createColumn({
16     fieldId: 'id'
17   }),
18   mySalesRepJoin.createColumn({
19     fieldId: 'entityid'
20   }),
21   mySalesRepJoin.createColumn({
22     fieldId: 'email'
23   }),
24   mySalesRepJoin.createColumn({
25     fieldId: 'hiredate'
26   })
27 ];
28
29 myCustomerQuery.sort = [
30   myCustomerQuery.createSort({
31     column: myCustomerQuery.columns[1]
32   }),
33   mySalesRepJoin.createSort({
34     column: mySalesRepJoin.columns[0],
35     ascending: false
36   })
];

```

```

37 ];
38
39 var resultSet = myCustomerQuery.run();
40 ...
41 //Add additional code

```

query.createPeriod(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a query.Period object.
Returns	query.Period
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2020.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.code	string	required	The code of the period to create. Use a value from the query.PeriodCode enum for this parameter.
options.adjustment	string	optional	The adjustment of the period to create. Use a value from the query.PeriodAdjustment enum for this parameter. The default value is query.PeriodAdjustment.NOT_LAST .
options.type	string	optional	The type of the period to create. Use a value from the query.PeriodType enum for this parameter. The default value is query.PeriodType.START .

Errors

Error Code	Thrown If
INVALID_PERIOD_ADJUSTMENT	The specified period adjustment is not a value from the query.PeriodAdjustment enum.

Error Code	Thrown If
INVALID_PERIOD_CODE	The specified period code is not a value from the query.PeriodCode enum.
INVALID_PERIOD_TYPE	The specified period type is not a value from the query.PeriodType enum.
WRONG_PARAMETER_TYPE	Any of the parameters is not a string.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPeriod = query.createPeriod({
4   code: query.PeriodCode.LAST_PERIOD,
5   adjustment: query.PeriodAdjustment.ALL,
6   type: query.PeriodType.END
7 });
8
9 //myQuery is an existing query.Query object
10 var myComplexCondition = myQuery.createCondition({
11   fieldId: 'trandate',
12   operator: query.Operator.BEFORE,
13   values: [myPeriod]
14 });
15 ...
16 // Add additional code

```

query.createRelativeDate(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a query.RelativeDate object that represents a date relative to the current date. Use this method to create a query.RelativeDate object to use as part of a query condition. After you create a query.RelativeDate object, you can use it directly in the values parameter of Query.createCondition(options) or Component.createCondition(options).</p> <p>When you call this method, the options.dateId parameter determines the relative date that is created. The options.dateId parameter uses values from the query.DateId enum, and these values describe potential dates relative to the current date. Use them along with the options.value parameter to create a relative date. For example, to create a relative date that represents the date three weeks before the current date, call <code>query.createRelativeDate(options)</code> with an options.dateId value of <code>query.DateId.WEEKS_AGO</code> and an options.value value of 3. To create a relative date that represents the date three weeks after the current date, call <code>query.createRelativeDate(options)</code> with an options.dateId value of <code>query.DateId.WEEKS_FROM_NOW</code> and an options.value value of 3.</p>
Returns	query.RelativeDate
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2019.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.dateId	string	required	The ID of the relative date to create. Use the query.DateId enum to set this value.
options.value	number	required	The value to use to create the relative date. This value depends on the value that you specify for options.dateId. For example, to create a relative date that represents the date five days before the current date, use an options.value value of 5 and an options.dateId value of query.DateId.DAYS_AGO.

Errors

Error Code	Thrown If
INVALID_DATE_ID	The specified value for options.dateId is not a value from the query.DateId enum.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myRelativeDate = query.createRelativeDate({
8   dateId: query.DateId.DAYS_AGO,
9   value: 5
10});
11
12 myTransactionQuery.condition = myTransactionQuery.createCondition({
13   fieldId: 'trandate',
14   operator: query.Operator.WITHIN,
15   values: [query.RelativeDateRange.THREE_FISCAL_YEARS_AGO.start, myRelativeDate]
16 });
17 ...
18 // Add additional code

```

query.delete(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Deletes an existing query.</p> <p>Use this method to delete a query definition that was previously created using the SuiteAnalytics Workbook UI. After the query is deleted, it is no longer available and cannot be modified or executed.</p> <p> Important: The N/query module lets you create and run queries using the SuiteAnalytics Workbook query engine. You can use the N/query module to load and delete existing queries, but you cannot save queries. You can save queries using the SuiteAnalytics Workbook interface. For more information, see the help topic Navigating SuiteAnalytics Workbook.</p>
Returns	void
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	5 units
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The script ID of the query to delete.

Errors

Error Code	Thrown If
UNABLE_TO_DELETE_QUERY	A query with the specified ID cannot be deleted because the query does not exist or you do not have permission to delete it.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 query.delete({
4   id: 'custworkbook237'
5 });
6 ...
7 // Add additional code

```

query.listTables(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Lists the table view objects that are included in a workbook in SuiteAnalytics Workbook.</p> <p>This method returns an array of Objects. Each Object represents a table view and includes the following properties with associated values:</p> <ul style="list-style-type: none"> ■ name — The name of the table view object ■ scriptId — The script ID of the table view object
Returns	Array<Object>
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2020.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.workbookId	string	required	The ID of the workbook containing the table view objects to list.

Errors

Error Code	Thrown If
MISSING_REQD_ARGUMENT	A required parameter is missing.
SCRIPT_ID_OF_WORKBOOK_IS_REQUIRED	The specified workbook ID represents an analytical record that is not a workbook.
SSS_INVALID_SCRIPT_ID_1	The specified workbook ID is not valid.

Error Code	Thrown If
WRONG_PARAMETER_TYPE	The specified workbook ID is not a string.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var tables = query.listTables({
4   workbookId: 'custworkbook237'
5 });
6 ...
7 // Add additional code

```

query.load(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Lets you load an existing query definition as a query.Query object. Use this method to load a query definition that was previously created using the SuiteAnalytics Workbook UI. After the query is loaded, you can modify the query definition (for example, by setting additional property values), join the query definition with other query types, and execute the query in the same way as queries that you create using query.create(options). You can provide either the workbook ID or the dataset ID of the query definition to load. You can obtain these IDs in the SuiteAnalytics Workbook UI. For more information, see the help topic Navigating SuiteAnalytics Workbook. This method loads only the table view in the specified workbook. However, a workbook can include no table views, or it can include multiple table views. This method throws an exception in each of these cases (see Errors). If a workbook includes multiple table views, you can use query.listTables(options) to determine the ID of a specific table view. You can use this ID to load the corresponding table view.</p> <p> Important: The N/query module lets you create and run queries using the SuiteAnalytics Workbook query engine. You can use the N/query module to load and delete existing queries, but you cannot save queries. You can save queries using the SuiteAnalytics Workbook interface. For more information, see the help topic Navigating SuiteAnalytics Workbook.</p>
Returns	query.Query
Supported Script Types	<p>Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	5 units
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The workbook ID or dataset ID of the query definition to load. You can obtain these IDs in the SuiteAnalytics Workbook UI. For more information, see the help topic Navigating SuiteAnalytics Workbook .

Errors

Error Code	Thrown If
UNABLE_TO_LOAD_QUERY	A query with the specified ID cannot be loaded because the query does not exist or you do not have permission to load it.
WORKBOOK_MORE_TABLEVIEWS_ARE_ASSIGNED	More than one table view is included in the specified workbook or dataset.
WORKBOOK_NO_TABLEVIEW_IS_ASSIGNED	No table views are included in the specified workbook or dataset.

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myLoadedQuery = query.load({
4   id: 'custworkbook237'
5 });
6
7 var mySalesRepJoin = myLoadedQuery.autoJoin({
8   fieldId: 'salesrep'
9 });
10
11 var results = myLoadedQuery.run();
12 ...
13 // Add additional code

```

query.load.promise(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Loads an existing query asynchronously as a query.Query object. Use this method to asynchronously load a query definition that was previously created using the SuiteAnalytics Workbook UI. After the query is loaded, you can modify the query definition (for example, by setting additional property values), join the query definition with other query types, and execute the query in the same way as queries that you create using query.create(options) .
---------------------------	--

	<p>You can provide either the workbook ID or the dataset ID of the query definition to load. You can obtain these IDs in the SuiteAnalytics Workbook UI. For more information, see the help topic Navigating SuiteAnalytics Workbook.</p> <p>This method loads only the table view in the specified workbook. However, a workbook can include no table views, or it can include multiple table views. This method throws an exception in each of these cases (see Errors). If a workbook includes multiple table views, you can use <code>query.listTables(options)</code> to determine the ID of a specific table view. You can use this ID to load the corresponding table view.</p>
	<p> Important: The N/query module lets you create and run queries using the SuiteAnalytics Workbook query engine. You can use the N/query module to load and delete existing queries, but you cannot save queries. You can save queries using the SuiteAnalytics Workbook interface. For more information, see the help topic Navigating SuiteAnalytics Workbook.</p>
	<p> Note: The parameters and errors thrown for this method are the same as those for <code>query.load(options)</code>. For more information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	<code>query.load(options)</code>
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	5 units
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description
options.id	string	required	<p>The workbook ID or dataset ID of the query definition to load.</p> <p>You can obtain these IDs in the SuiteAnalytics Workbook UI. For more information, see the help topic Navigating SuiteAnalytics Workbook.</p>

Errors

Error Code	Thrown If
UNABLE_TO_LOAD_QUERY	A query with the specified ID cannot be loaded because the query does not exist or you do not have permission to load it.

Error Code	Thrown If
WORKBOOK_MORE_TABLEVIEWS_ARE_ASSIGNED	More than one table view is included in the specified workbook or dataset.
WORKBOOK_NO_TABLEVIEW_IS_ASSIGNED	No table views are included in the specified workbook or dataset.

query.runSuiteQL(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Runs an arbitrary SuiteQL query.</p> <p>SuiteQL is a query language based on the SQL-92 revision of the SQL database query language. It provides advanced query capabilities you can use to access your NetSuite records and data.</p> <p>You can specify the SuiteQL query as one of the following:</p> <ul style="list-style-type: none"> ■ A string representation of the SuiteQL query
	<pre> 1 var results = query.runSuiteQL({ 2 query: 'SELECT customer.entityid, customer.email FROM customer' 3 }); </pre>
	<ul style="list-style-type: none"> ■ A query.SuiteQL object <pre> 1 //In this example, mySuiteQLCustomerQuery is an existing SuiteQL object 2 var results = query.runSuiteQL(mySuiteQLCustomerQuery); </pre> <ul style="list-style-type: none"> ■ A JavaScript Object that contains a query property and, optionally, a params property <pre> 1 var results = query.runSuiteQL({ 2 query: 'SELECT customer.entityid, customer.email FROM customer WHERE customer.isperson = ?', 3 params: [true] 4 }); </pre>
	<p>Note: If the SuiteAnalytics Connect feature is enabled in your NetSuite account, there is no limit to the number of results this method can return. If the SuiteAnalytics Connect feature is not enabled, this method can return a maximum of 100,000 results. For more information about SuiteAnalytics Connect, see the help topic SuiteAnalytics Connect.</p>
Returns	<p>query.ResultSet</p>
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	<p>10 units</p>
Module	<p>N/query Module</p>
Sibling Module Members	<p>N/query Module Members</p>

Since	2020.1
-------	--------

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.query	string	required	The string representation of the SuiteQL query to run.
options.params	Array<string number boolean>	optional	The parameters to use in the SuiteQL query.

Errors

Error Code	Thrown If
MISSING_REQD_ARGUMENT	The parameter is missing.
SSS_INVALID_TYPE_ARG	Types other than string, number, or boolean are included in the options.params array.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var results = query.runSuiteQL({
4   query: 'SELECT customer.entityid, customer.email FROM customer WHERE customer.isperson = ?',
5   params: [true]
6 });
7 ...
8 // Add additional code

```

query.runSuiteQLPaged(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Runs an arbitrary SuiteQL query as a paged query.</p> <p>SuiteQL is a query language based on the SQL-92 revision of the SQL database query language. It provides advanced query capabilities you can use to access your NetSuite records and data.</p> <p>You can specify the SuiteQL query as one of the following:</p> <ul style="list-style-type: none"> ■ A string representation of the SuiteQL query <pre> 1 var results = query.runSuiteQLPaged({ 2 query: 'SELECT customer.entityid, customer.email FROM customer', 3 pageSize: 10 4 }); </pre>
---------------------------	--

	<ul style="list-style-type: none"> ■ A query.SuiteQL object <pre> 1 // To use this approach, you must load the N/util module in your script 2 // In this example, mySuiteQLCustomerQuery is an existing SuiteQL object 3 var pageSizeObject = { 4 pageSize: 10 5 }; 6 var results = query.runSuiteQL(util.extend(mySuiteQLCustomerQuery, pageSizeObject)); </pre> <p>Although this approach is supported, you should create a query.SuiteQL object and call SuiteQL.runPaged(options), if possible, to run a paged SuiteQL query.</p> <ul style="list-style-type: none"> ■ A JavaScript Object that contains a query property and, optionally, a params property <pre> 1 var results = query.runSuiteQL({ 2 query: 'SELECT customer.entityid, customer.email FROM customer WHERE customer.isperson = ?', 3 params: [true], 4 pageSize: 10 5 }); </pre>
	<p>Note: If the SuiteAnalytics Connect feature is enabled in your NetSuite account, there is no limit to the number of results this method can return. If the SuiteAnalytics Connect feature is not enabled, this method can return a maximum of 100,000 results across all pages in the result set. For more information about SuiteAnalytics Connect, see the help topic SuiteAnalytics Connect.</p>
	<p>For more information and examples of using SuiteQL in the N/query module, see SuiteQL in the N/query Module. For more information about SuiteQL in general, see the help topic SuiteQL.</p>
Returns	query.PagedData
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2020.1

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description
options.query	string	required	The string representation of the SuiteQL query to run.
options.params	Array<string number boolean>	optional	The parameters to use in the SuiteQL query.
options.pageSize	number	optional	The size of each page in the query results. The default value is 50 results per page. The minimum page size is 5 results per page, and the maximum page size is 1000 results per page.

Errors

Error Code	Thrown If
MISSING_REQD_ARGUMENT	The parameter is missing.
SSS_INVALID_TYPE_ARG	Types other than string, number, or boolean are included in the options.params array.

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var results = query.runSuiteQLPaged({
4     query: 'SELECT customer.entityid, customer.email FROM customer WHERE customer.isperson = ?',
5     params: [true],
6     pageSize: 20
7 });
8 ...
9 // Add additional code

```

query.Aggregate

i Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds the string values for aggregate functions supported with the N/query Module. An aggregate function performs a calculation on the column or condition values and returns a single value.</p> <p>Each value in this enum (except MEDIAN) has two variants: distinct (using the _DISTINCT suffix) and nondistinct (using no suffix). The variant determines whether the aggregate function operates on all instances of duplicate values or on just a single instance of the value. For example, consider a situation in which the MAXIMUM aggregate function is used to determine the maximum of a set of values. When using the distinct variant (MAXIMUM_DISTINCT), the aggregate function considers each instance of duplicate values. So if the set of values includes three distinct values that are all equal and all represent the maximum value in the set, the aggregate function lists all three instances. When using the nondistinct variant (MAXIMUM), only one instance of the maximum value is listed, regardless of the number of instances of that maximum value in the set.</p> <p>This enum is used to pass the aggregate function argument to Component.createColumn(options), Component.createCondition(options), Query.createColumn(options), and Query.createCondition(options).</p> <p>i Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.1

Values

Value	Description
AVERAGE	Calculates the average value.
AVERAGE_DISTINCT	Calculates the average distinct value.
COUNT	Counts the number of results.
COUNT_DISTINCT	Counts the number of distinct results.
MAXIMUM	Determines the maximum value. If the values are dates, the most recent date is determined.
MAXIMUM_DISTINCT	Determines the maximum distinct value. If the values are dates, the most recent date is determined.
MEDIAN	Calculates the median value.
MINIMUM	Determines the minimum value. If the values are dates, the earliest date is determined.
MINIMUM_DISTINCT	Determines the minimum distinct value. If the values are dates, the earliest date is determined.
SUM	Adds all values.
SUM_DISTINCT	Adds all distinct values.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myAggColumn = myTransactionQuery.createColumn({
8   fieldId: 'amount',
9   aggregate: query.Aggregate.AVERAGE
10});
11
12 myTransactionQuery.columns = [myAggColumn];
13 ...
14 // Add additional code

```

query.DateId



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported date codes in relative dates. This enum is used to pass the date ID argument to <code>query.createRelativeDate(options)</code> . It is also used as the value of the <code>RelativeDate.dateId</code> property. When <code>query.createRelativeDate(options)</code> is called, the enum value that you specify is set as the value of the <code>RelativeDate.dateId</code> property.
-------------------------	---

	<p>When creating a relative date using <code>query.createRelativeDate(options)</code>, use the values in this enum to specify a date relative to the current date. For example, to create a relative date that represents the date a certain number of days before the current date, use the <code>DateId.DAYS_AGO</code> enum value. To create a relative date that represents the date a certain number of months after the current date, use the <code>DateId.MONTHS_FROM_NOW</code> enum value.</p> <p>The values in this enum might look similar to the values in the <code>query.RelativeDateRange</code> enum, but each enum is used for a different purpose:</p> <ul style="list-style-type: none"> ■ Use <code>query.DateId</code> enum values to create a <code>query.RelativeDate</code> object using <code>query.createRelativeDate(options)</code>. After you create this object, you can use it in query conditions that you create using <code>Query.createCondition(options)</code> or <code>Component.createCondition(options)</code>. ■ Use <code>query.RelativeDateRange</code> enum values directly in query conditions that you create using <code>Query.createCondition(options)</code> or <code>Component.createCondition(options)</code>. Each value in the <code>query.RelativeDateRange</code> enum represents a date range, and you can use these values in the <code>values</code> parameter of <code>Query.createCondition(options)</code> or <code>Component.createCondition(options)</code>. <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p> </div>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2019.1

Values

Value	Sets <code>RelativeDate.dateId</code> Property To
DAYS_AGO	dago
DAYS_FROM_NOW	dfn
HOURS_AGO	hago
HOURS_FROM_NOW	hfn
MINUTES_AGO	nago
MINUTES_FROM_NOW	nfn
MONTHS_AGO	mago
MONTHS_FROM_NOW	mfn
QUARTERS_AGO	qago
QUARTERS_FROM_NOW	qfn
SECONDS_AGO	sago
SECONDS_FROM_NOW	sfn
WEEKS_AGO	wago
WEEKS_FROM_NOW	wfn

Value	Sets <code>RelativeDate.dateId</code> Property To
YEARS_AGO	yago
YEARS_FROM_NOW	yfn

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myRelativeDate = query.createRelativeDate({
4   dateId: query.DateId.DAYS_AGO,
5   value: 2
6 });
7 ...
8 // Add additional code

```

query.FieldContext

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for the field context to use when creating a column using Query.createColumn(options) or Component.createColumn(options) . The field context determines how field values are displayed in a column. For example, you can specify that a column should display raw data (such as internal IDs), consolidated or converted amounts (such as currency totals), or user-friendly values (such as names).  Note: JavaScript does not include an enumeration type. The SuiteScript documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2019.1

Values

Value	Description
CONVERTED	Displays converted currency amounts using the exchange rate that was in effect on a specific date.
CURRENCY_CONSOLIDATED	Displays consolidated currency amounts in the base currency.
DISPLAY	Displays user-friendly field values. For example, for the entity field on Transaction records, using the DISPLAY enum value displays the name of the entity instead of its ID.

Value	Description
HIERARCHY	Displays user-friendly field values for hierarchical fields (for example, "Parent Company : SUB CAD"). This value is similar to the DISPLAY enum value but applies to hierarchical fields.
HIERARCHY_IDENTIFIER	Displays raw field values for hierarchical fields (for example, "1 : 5"). This value is similar to the RAW enum value but applies to hierarchical fields.
RAW	Displays raw field values. For example, for the entity field on Transaction records, using the RAW enum value displays the ID of the entity.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myContextColumn = myTransactionQuery.createColumn({
8   fieldId: 'netamount',
9   context: query.FieldContext.CURRENCY_CONSOLIDATED
10 });
11 ...
12 // Add additional code

```

query.Operator



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds the string values for operators supported in SuiteScript Analytic APIs. SuiteScript Analytic APIs help you work with analytical data in NetSuite using SuiteScript. For more information, see the help topic SuiteScript Analytic APIs.</p> <p>This enum is used to specify the operator argument to Query.createCondition(options) and Component.createCondition(options).</p> <p>Values for this enum are listed in the Values section. The following columns provide information about each operator:</p> <ul style="list-style-type: none"> ■ Value — Use these values to specify operators in most queries (for example, <code>query.Operator.BEFORE</code>). To use these values, you must include the N/query module in your script. ■ Mapped String Value — Use these values as strings that represent the corresponding operators (for example, 'BEFORE'). To use these values, you do not need to include the N/query module in your script. ■ Use For — Use this column to determine the value types that each operator supports. For example, the <code>query.Operator.AFTER</code> operator is designed to be used with date/time values, and you cannot use this operator with string or boolean values. Some operators are similar but are designed to be used with different value types (such as <code>query.Operator.IS</code> for boolean values and <code>query.Operator.EQUAL</code> for numeric values).
------------------	---

	<p>For multi-select fields, you can use the <code>query.Operator.INCLUDE_*</code> and <code>query.Operator.EXCLUDE_*</code> values to specify a set of exact field values. For example, to obtain script deployment records that apply to all execution contexts except the WEBAPP and WEBSTORE contexts, you can use the <code>query.Operator.EXCLUDE_ALL</code> operator. You cannot use the <code>query.Operator.ANY_OF_NOT</code> operator to obtain these records, because this operator uses an implicit OR operator between all specified values. The <code>query.Operator.EXCLUDE_ALL</code> operator uses an implicit AND operator between all specified values, which lets you create more complex conditions.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.</p> </div>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.1

Values

Value	Mapped String Value	Use For
AFTER	AFTER	Date/time values
AFTER_NOT	AFTER_NOT	Date/time values
ANY_OF	ANY_OF	Single-select fields Multi-select fields
ANY_OF_NOT	ANY_OF_NOT	Single-select fields Multi-select fields
BEFORE	BEFORE	Date/time values
BEFORE_NOT	BEFORE_NOT	Date/time values
BETWEEN	BETWEEN	Date/time values
BETWEEN_NOT	BETWEEN_NOT	Date/time values
CONTAIN	CONTAIN	Strings
CONTAIN_NOT	CONTAIN_NOT	Strings
EMPTY	EMPTY	Single-select fields Multi-select fields
EMPTY_NOT	EMPTY_NOT	Single-select fields Multi-select fields
ENDWITH	ENDWITH	Strings
ENDWITH_NOT	ENDWITH_NOT	Strings
EQUAL	EQUAL	Numbers
EQUAL_NOT	EQUAL_NOT	Numbers

Value	Mapped String Value	Use For
EXCLUDE_ALL	MN_EXCLUDE	Multi-select fields
EXCLUDE_ANY	MN_EXCLUDE_ALL	Multi-select fields
EXCLUDE_EXACTLY	MN_EXCLUDE_EXACTLY	Multi-select fields
GREATER	GREATER	Numbers
GREATER_NOT	GREATER_NOT	Numbers
GREATER_OR_EQUAL	GREATER_OR_EQUAL	Numbers
GREATER_OR_EQUAL_NOT	GREATER_OR_EQUAL_NOT	Numbers
INCLUDE_ALL	MN_INCLUDE_ALL	Multi-select fields
INCLUDE_ANY	MN_INCLUDE	Multi-select fields
INCLUDE_EXACTLY	MN_INCLUDE_EXACTLY	Multi-select fields
IS	IS	Boolean values
IS_NOT	IS_NOT	Boolean values
LESS	LESS	Numbers
LESS_NOT	LESS_NOT	Numbers
LESS_OR_EQUAL	LESS_OR_EQUAL	Numbers
LESS_OR_EQUAL_NOT	LESS_OR_EQUAL_NOT	Numbers
ON	ON	Date/time values
ON_NOT	ON_NOT	Date/time values
ON_OR_AFTER	ON_OR_AFTER	Date/time values
ON_OR_AFTER_NOT	ON_OR_AFTER_NOT	Date/time values
ON_OR_BEFORE	ON_OR_BEFORE	Date/time values
ON_OR_BEFORE_NOT	ON_OR_BEFORE_NOT	Date/time values
START_WITH	START_WITH	Strings
START_WITH_NOT	START_WITH_NOT	Strings
WITHIN	WITHIN	Date/time values
WITHIN_NOT	WITHIN_NOT	Date/time values

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...

```

```

3  var myCustomerQuery = query.create({
4      type: query.Type.CUSTOMER
5  });
6
7  var mySalesRepJoin = myCustomerQuery.autoJoin({
8      fieldId: 'salesrep'
9  });
10
11 var firstCondition = myCustomerQuery.createCondition({
12     fieldId: 'id',
13     operator: query.Operator.EQUAL,
14     values: 107
15 });
16 var secondCondition = myCustomerQuery.createCondition({
17     fieldId: 'id',
18     operator: query.Operator.EQUAL,
19     values: 2647
20 });
21 var thirdCondition = mySalesRepJoin.createCondition({
22     fieldId: 'email',
23     operator: query.Operator.START_WITH_NOT,
24     values: 'foo'
25 });
26
27 myCustomerQuery.condition = myCustomerQuery.and(
28     thirdCondition, myCustomerQuery.not(
29         myCustomerQuery.or(firstCondition, secondCondition)
30     )
31 );
32
33 var resultSet = myCustomerQuery.run();
34 ...
35 // Add additional code

```

query.PeriodAdjustment



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds the string values for adjustment types for a period. This enum is used to pass the adjustment argument to query.createPeriod(options).</p>
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2020.1

Values

Value
ALL

Value
NOT_LAST

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPeriod = query.createPeriod({
4   code: query.PeriodCode.LAST_PERIOD,
5   adjustment: query.PeriodAdjustment.ALL,
6   type: query.PeriodType.END
7 });
8 ...
9 // Add additional code

```

query.PeriodCode



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for period codes for a period. This enum is used to pass the code argument to query.createPeriod(options) . Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2020.1

Values

Value	Sets Period.code Property To
FIRST_FISCAL_QUARTER_LAST_FY	Q1LFY
FIRST_FISCAL_QUARTER_THIS_FY	Q1TFY
FISCAL_QUARTER_BEFORE_LAST	QBL
FISCAL_YEAR_BEFORE_LAST	FYBL
FOURTH_FISCAL_QUARTER_LAST_FY	Q4LFY
FOURTH_FISCAL_QUARTER_THIS_FY	Q4TFY
LAST_FISCAL_QUARTER	LQ

Value	Sets Period.code Property To
LAST_FISCAL_QUARTER_ONE_FISCAL_YEAR_AGO	LQOLFY
LAST_FISCAL_QUARTER_TO_PERIOD	LFQTP
LAST_FISCAL_YEAR	LFY
LAST_FISCAL_YEAR_TO_PERIOD	LFYTP
LAST_PERIOD	LP
LAST_PERIOD_ONE_FISCAL_QUARTER_AGO	LPOLQ
LAST_PERIOD_ONE_FISCAL_YEAR_AGO	LPOLFY
LAST_ROLLING_18_PERIODS	LR18FP
LAST_ROLLING_6_FISCAL_QUARTERS	LR6FQ
PERIOD_BEFORE_LAST	PBL
SAME_FISCAL_QUARTER_LAST_FY	TQOLFY
SAME_FISCAL_QUARTER_LAST_FY_TO_PERIOD	TFQOLFYTP
SAME_PERIOD_LAST_FY	TPOLFY
SAME_PERIOD_LAST_FISCAL_QUARTER	TPOLQ
SECOND_FISCAL_QUARTER_LAST_FY	Q2LFY
SECOND_FISCAL_QUARTER_THIS_FY	Q2TFY
THIRD_FISCAL_QUARTER_LAST_FY	Q3LFY
THIRD_FISCAL_QUARTER_THIS_FY	Q3TFY
THIS_FISCAL_QUARTER	TQ
THIS_FISCAL_QUARTER_TO_PERIOD	TFQTP
THIS_FISCAL_YEAR	TFY
THIS_FISCAL_YEAR_TO_PERIOD	TFYTP
THIS_PERIOD	TP

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPeriod = query.createPeriod({
4   code: query.PeriodCode.LAST_PERIOD,
5   adjustment: query.PeriodAdjustment.ALL,
6   type: query.PeriodType.END
7 });
8 ...
9 // Add additional code

```

query.PeriodType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds the string values for period types for a period. This enum is used to pass the type argument to query.createPeriod(options).</p> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2020.1

Values

Value
END
START

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myPeriod = query.createPeriod({
4   code: query.PeriodCode.LAST_PERIOD,
5   adjustment: query.PeriodAdjustment.ALL,
6   type: query.PeriodType.END
7 });
8 ...
9 // Add additional code

```

query.RelativeDateRange



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds query.RelativeDate object values for supported date ranges in relative dates. This enum is used to pass the values argument to Query.createCondition(options) and Component.createCondition(options). It is also used as the value of the RelativeDate.value property. Each value in this enum represents a date range. When Query.createCondition(options) or Component.createCondition(options) is called with a query.RelativeDate object as the values argument, this object is set as the value of the RelativeDate.value property.</p>
-------------------------	---

	<p>When creating a condition using Query.createCondition(options) or Component.createCondition(options), use the values in this enum (along with values in the query.Operator enum) to specify a range of dates relative to the current date. For example, to create a condition to match dates that occur before the current date, use the query.RelativeDateRange.TODAY enum value and the query.Operator.BEFORE enum value. To create a condition to match dates that occur after last year, use the query.RelativeDateRange.LAST_YEAR enum value and the query.Operator.AFTER enum value. For more information about relative dates, see Relative Dates in the N/query Module.</p> <p>The values in this enum might look similar to the values in the query.DateId enum, but each enum is used for a different purpose:</p> <ul style="list-style-type: none"> ■ Use query.DateId enum values to create a query.RelativeDate object using query.createRelativeDate(options). After you create this object, you can use it in query conditions that you create using Query.createCondition(options) or Component.createCondition(options). ■ Use query.RelativeDateRange enum values directly in query conditions that you create using Query.createCondition(options) or Component.createCondition(options). Each value in the query.RelativeDateRange enum represents a date range, and you can use these values in the values parameter of Query.createCondition(options) or Component.createCondition(options). <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2019.1

Values

Value	RelativeDate.dateId Property
FISCAL_HALF_BEFORE_LAST	FHBL
FISCAL_HALF_BEFORE_LAST_TO_DATE	FHBLTD
FISCAL_QUARTER_BEFORE_LAST	FQBL
FISCAL_QUARTER_BEFORE_LAST_TO_DATE	FQBLTD
FISCAL_YEAR_BEFORE_LAST	FYBL
FISCAL_YEAR_BEFORE_LAST_TO_DATE	FYBLTD
FIVE_DAYS_AGO	DAGO5
FIVE_DAYS_FROM_NOW	DFN5
FOUR_DAYS_AGO	DAGO4
FOUR_DAYS_FROM_NOW	DFN4
FOUR_WEEKS_STARTING_THIS_WEEK	TWN3W
LAST_BUSINESS_WEEK	LBW

Value	RelativeDate.dateId Property
LAST_FISCAL_HALF	LFH
LAST_FISCAL_HALF_ONE_FISCAL_YEAR_AGO	LFHLFY
LAST_FISCAL_HALF_TO_DATE	LFHTD
LAST_FISCAL_QUARTER	LFQ
LAST_FISCAL_QUARTER_ONE_FISCAL_YEAR_AGO	LFQLFY
LAST_FISCAL_QUARTER_TO_DATE	LFQTD
LAST_FISCAL_QUARTER_TWO_FISCAL_YEARS_AGO	LFQFYBL
LAST_FISCAL_YEAR	LFY
LAST_FISCAL_YEAR_TO_DATE	LFYTD
LAST_MONTH	LM
LAST_MONTH_ONE_FISCAL_QUARTER_AGO	LMLFQ
LAST_MONTH_ONE_FISCAL_YEAR_AGO	LMLFY
LAST_MONTH_TO_DATE	LMTD
LAST_MONTH_TWO_FISCAL_QUARTERS_AGO	LMFQBL
LAST_MONTH_TWO_FISCAL_YEARS_AGO	LMFYBL
LAST_ROLLING_HALF	LRH
LAST_ROLLING_QUARTER	LRQ
LAST_ROLLING_YEAR	LRY
LAST_WEEK	LW
LAST_WEEK_TO_DATE	LWTD
LAST_YEAR	LY
LAST_YEAR_TO_DATE	LYTD
MONTH_AFTER_NEXT	MAN
MONTH_AFTER_NEXT_TO_DATE	MANTD
MONTH_BEFORE_LAST	MBL
MONTH_BEFORE_LAST_TO_DATE	MBLTD
NEXT_BUSINESS_WEEK	NBW
NEXT_FISCAL_HALF	NFH
NEXT_FISCAL_QUARTER	NFQ
NEXT_FISCAL_YEAR	NFY
NEXT_FOUR_WEEKS	N4W
NEXT_MONTH	NM

Value	RelativeDate.dateId Property
NEXT_ONE_HALF	NOH
NEXT_ONE_MONTH	NOM
NEXT_ONE_QUARTER	NOQ
NEXT_ONE_WEEK	NOW
NEXT_ONE_YEAR	NOY
NEXT_WEEK	NW
NINETY_DAYS_AGO	DAGO90
NINETY_DAYS_FROM_NOW	DFN90
ONE_YEAR_BEFORE_LAST	OYBL
PREVIOUS_FISCAL_QUARTERS_LAST_FISCAL_YEAR	PQLFY
PREVIOUS_FISCAL_QUARTERS_THIS_FISCAL_YEAR	PQTFY
PREVIOUS_MONTHS_LAST_FISCAL_HALF	PMLFH
PREVIOUS_MONTHS_LAST_FISCAL_QUARTER	PMLFQ
PREVIOUS_MONTHS_LAST_FISCAL_YEAR	PMLFY
PREVIOUS_MONTHS_SAME_FISCAL_HALF_LAST_FISCAL_YEAR	PMSFHLFY
PREVIOUS_MONTHS_SAME_FISCAL_QUARTER_LAST_FISCAL_YEAR	PMSFQLFY
PREVIOUS_MONTHS_THIS_FISCAL_HALF	PMTFH
PREVIOUS_MONTHS_THIS_FISCAL_QUARTER	PMTFQ
PREVIOUS_MONTHS_THIS_FISCAL_YEAR	PMTFY
PREVIOUS_ONE_DAY	OD
PREVIOUS_ONE_HALF	OH
PREVIOUS_ONE_MONTH	OM
PREVIOUS_ONE_QUARTER	OQ
PREVIOUS_ONE_WEEK	OW
PREVIOUS_ONE_YEAR	OY
PREVIOUS_ROLLING_HALF	PRH
PREVIOUS_ROLLING_QUARTER	PRQ
PREVIOUS_ROLLING_YEAR	PRY
SAME_DAY_FISCAL_QUARTER_BEFORE_LAST	SDFQBL
SAME_DAY_FISCAL_YEAR_BEFORE_LAST	SDFYBL
SAME_DAY_LAST_FISCAL_QUARTER	SDLFQ
SAME_DAY_LAST_FISCAL_YEAR	SDLFY

Value	RelativeDate.dateId Property
SAME_DAY_LAST_MONTH	SDLM
SAME_DAY_LAST_WEEK	SDLW
SAME_DAY_MONTH_BEFORE_LAST	SDMBL
SAME_DAY_WEEK_BEFORE_LAST	SDWBL
SAME_FISCAL_HALF_LAST_FISCAL_YEAR	SFHLFY
SAME_FISCAL_HALF_LAST_FISCAL_YEAR_TO_DATE	SFHLFYTD
SAME_FISCAL_QUARTER_FISCAL_YEAR_BEFORE_LAST	SFQFYBL
SAME_FISCAL_QUARTER_LAST_FISCAL_YEAR	SFQLFY
SAME_FISCAL_QUARTER_LAST_FISCAL_YEAR_TO_DATE	SFQLFYTD
SAME_MONTH_FISCAL_QUARTER_BEFORE_LAST	SMFQBL
SAME_MONTH_FISCAL_YEAR_BEFORE_LAST	SMFYBL
SAME_MONTH_LAST_FISCAL_QUARTER	SMLFQ
SAME_MONTH_LAST_FISCAL_QUARTER_TO_DATE	SMLFQTD
SAME_MONTH_LAST_FISCAL_YEAR	SMLFY
SAME_MONTH_LAST_FISCAL_YEAR_TO_DATE	SMLFYTD
SAME_WEEK_FISCAL_YEAR_BEFORE_LAST	SWFYBL
SAME_WEEK_LAST_FISCAL_YEAR	SWLFY
SIXTY_DAYS_AGO	DAGO60
SIXTY_DAYS_FROM_NOW	DFN60
TEN_DAYS_AGO	DAGO10
TEN_DAYS_FROM_NOW	DFN10
THIRTY_DAYS_AGO	DAGO30
THIRTY_DAYS_FROM_NOW	DFN30
THIS_BUSINESS_WEEK	TBW
THIS_FISCAL_HALF	TFH
THIS_FISCAL_HALF_TO_DATE	TFHTD
THIS_FISCAL_QUARTER	TFQ
THIS_FISCAL_QUARTER_TO_DATE	TFQTD
THIS_FISCAL_YEAR	TFY
THIS_FISCAL_YEAR_TO_DATE	TFYTD
THIS_MONTH	TM
THIS_MONTH_TO_DATE	TMTD

Value	RelativeDate.dateId Property
THIS_ROLLING_HALF	TRH
THIS_ROLLING_QUARTER	TRQ
THIS_ROLLING_YEAR	TRY
THIS_WEEK	TW
THIS_WEEK_TO_DATE	TWTD
THIS_YEAR	TY
THIS_YEAR_TO_DATE	TYTD
THREE_DAYS_AGO	DAGO3
THREE_DAYS_FROM_NOW	DFN3
THREE_FISCAL_QUARTERS_AGO	FQB
THREE_FISCAL_QUARTERS_AGO_TO_DATE	FQBTD
THREE_FISCAL_YEARS_AGO	FYB
THREE_FISCAL_YEARS_AGO_TO_DATE	FYBD
THREE_MONTHS_AGO	MB
THREE_MONTHS_AGO_TO_DATE	MBTD
TODAY	TODAY
TODAY_TO_END_OF_THIS_MONTH	TODAYTTM
TOMORROW	TOMORROW
TWO_DAYS_AGO	DAGO2
TWO_DAYS_FROM_NOW	DFN2
WEEK_AFTER_NEXT	WAN
WEEK_AFTER_NEXT_TO_DATE	WANTD
WEEK_BEFORE_LAST	WBL
WEEK_BEFORE_LAST_TO_DATE	WBLTD
YESTERDAY	YESTERDAY

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6

```

```

7 | myTransactionQuery.condition = myTransactionQuery.createCondition({
8 |   fieldId: 'trandate',
9 |   operator: query.Operator.BEFORE,
10|   values: query.RelativeDateRange.TODAY
11| });
12| ...
13| // Add additional code

```

query.ReturnType

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds the string values for the formula return types supported with the N/query Module.</p> <p>This enum is used to pass the formula return type argument to Query.createColumn(options), Component.createColumn(options), Query.createCondition(options), and Component.createCondition(options).</p> <p>For more information about these return types, see the help topic Formula Fields.</p> <p>For more information on formulas, see the help topics SuiteAnalytics Workbook Overview, SQL Expressions, and Search Formula Examples and Tips.</p> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.</p>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.1

Values

Value
ANY
BOOLEAN
CLOBTEXT
CURRENCY
DATE
DATETIME
DURATION
FLOAT
INTEGER
KEY
PERCENT

Value
RELATIONSHIP
STRING
UNKNOWN

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myTransactionQuery = query.create({
4   type: query.Type.TRANSACTION
5 });
6
7 var myFormulaColumn = myTransactionQuery.createColumn({
8   type: query.ReturnType.CURRENCY,
9   formula: '{amount} * 125'
10 });
11 ...
12 // Add additional code

```

query.SortLocale



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for sort locales supported with the N/query Module . This enum is used to pass the locale argument to Query.createSort(options) and Component.createSort(options) .
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.2

Values

Value
ARABIC
ARABIC_ABJ_MATCH
ARABIC_ABJ_MATCH_CI

Value
ARABIC_ABJ_SORT
ARABIC_ABJ_SORT_CI
ARABIC_CI
ARABIC_MATCH
ARABIC_MATCH_CI
ASCII7
ASCII7_CI
AZERBAIJANI
AZERBAIJANI_CI
BENGALI
BENGALI_CI
BIG5
BIG5_CI
BINARY
BINARY_CI
BULGARIAN
BULGARIAN_CI
CANADIAN_M
CATALAN
CATALAN_CI
CROATIAN
CROATIAN_CI
CS_CZ
CZECH
CZECH_CI
CZECH_PUNCTUATION
CZECH_PUNCTUATION_CI
DANISH
DANISH_CI
DANISH_M
DA_DK
DE_DE

Value
DUTCH
DUTCH_CI
EBCDIC
EBCDIC_CI
EEC_EURO
EEC_EUROPA3
EEC_EUROPA3_CI
EEC_EURO_CI
EN
EN_AU
EN_CA
EN_GB
EN_US
ESTONIAN
ESTONIAN_CI
ES_AR
ES_ES
FINNISH
FINNISH_CI
FRENCH
FRENCH_AI
FRENCH_CI
FRENCH_M
FR_CA
FR_FR
GBK
GBK_AI
GBK_CI
GENERIC_M
GERMAN
GERMAN_AI
GERMAN_CI

Value
GERMAN_DIN
GERMAN_DIN_AI
GERMAN_DIN_CI
GREEK
GREEK_AI
GREEK_CI
HEBREW
HEBREW_AI
HEBREW_CI
HE_IL
HKSCS
HKSCS_AI
HKSCS_CI
HUNGARIAN
HUNGARIAN_AI
HUNGARIAN_CI
ICELANDIC
ICELANDIC_AI
ICELANDIC_CI
ID_ID
INDONESIAN
INDONESIAN_AI
INDONESIAN_CI
ITALIAN
ITALIAN_AI
ITALIAN_CI
IT_IT
JAPANESE_M
JA_JP
KOREAN_M
KO_KR
LATIN

Value
LATIN_AI
LATIN_CI
LATVIAN
LATVIAN_AI
LATVIAN_CI
LITHUANIAN
LITHUANIAN_AI
LITHUANIAN_CI
MALAY
MALAY_AI
MALAY_CI
NL_NL
NO_NO
NORWEGIAN
NORWEGIAN_AI
NORWEGIAN_CI
POLISH
POLISH_AI
POLISH_CI
PT_BR
PUNCTUATION
PUNCTUATION_AI
PUNCTUATION_CI
ROMANIAN
ROMANIAN_AI
ROMANIAN_CI
RUSSIAN
RUSSIAN_AI
RUSSIAN_CI
RU_RU
SCHINESE_PINYIN_M
SCHINESE_RADICAL_M

Value
SCHINESE_STROKE_M
SLOVAK
SLOVAK_AI
SLOVAK_CI
SLOVENIAN
SLOVENIAN_AI
SLOVENIAN_CI
SPANISH
SPANISH_AI
SPANISH_CI
SPANISH_M
SV_SE
SWEDISH
SWEDISH_AI
SWEDISH_CI
SWISS
SWISS_AI
SWISS_CI
TCHINESE_RADICAL_M
TCHINESE_STROKE_M
THAI_M
TH_TH
TR_TR
TURKISH
TURKISH_AI
TURKISH_CI
UKRAINIAN
UKRAINIAN_AI
UKRAINIAN_CI
UNICODE_BINARY
UNICODE_BINARY_AI
UNICODE_BINARY_CI

Value
VIETNAMESE
VIETNAMESE_AI
VIETNAMESE_CI
WEST_EUROPEAN
WEST_EUROPEAN_AI
WEST_EUROPEAN_CI
ZH_CN
ZH_TW

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myCustomerQuery = query.create({
4   type: query.Type.CUSTOMER
5 });
6
7 myCustomerQuery.columns = [
8   myCustomerQuery.createColumn({
9     fieldId: 'entityid'
10   })
11 ];
12
13 myCustomerQuery.sort = [
14   myCustomerQuery.createSort({
15     column: myCustomerQuery.columns[0],
16     locale: query.SortLocale.EN_CA
17   })
18 ];
19 ...
20 // Add additional code

```

query.Type

ⓘ Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for query types used in the query definition. This enum is used to pass the initial query type argument to query.create(options) .
-------------------------	--

	<p>Important: The N/query module supports the same record types that are supported by the SuiteAnalytics Workbook interface. For more information, see the help topic Available Record Types.</p>
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Module	N/query Module
Sibling Module Members	N/query Module Members
Since	2018.1

Values

Note: Before using these values, consider the following:
<ul style="list-style-type: none"> ■ A query type is not the same as a record type. The supported query types listed below do not necessarily correspond with the supported record types listed in the N/record Module. ■ Depending on your account, role, and enabled features, some of these values may not be available. ■ Custom record types are not included in this enum.

Enum Value	Sets Query.type Property To
ACCOUNT	account
ACCOUNTING_BOOK	accountingbook
ACCOUNTING_CONTEXT	accountingcontext
ACCOUNTING_PERIOD	accountingperiod
ALLOCATION_METHOD	allocationmethod
AMORTIZATION_SCHEDULE	amortizationschedule
AMORTIZATION_TEMPLATE	amortizationtemplate
AUTHORIZATION_CONSENT	authorizationconsent
BILLING_CLASS	billingclass
BILLING_RATE_CARD	billingratecard
BILLING_SCHEDULE	billingschedule
BILL_OF_DISTRIBUTION	billofdistribution
BILL_RUN	billrun
BILL_RUN_SCHEDULE	billrunschedule

Enum Value	Sets Query.type Property To
BIN	bin
BOM	bom
BOM_REVISION	bomrevision
BOM_REVISION_COMPONENT	bomrevisioncomponent
BUDGETCATEGORY	budgetcategory
BUDGETS	budgets
BUDGET_EXCHANGE_RATE	budgetexchangerate
BULK_PROC_SUBMISSION	bulkprobsubmission
BUNDLE_INSTALLATION_SCRIPT	bundleinstallationscript
BUNDLE_INSTALLATION_SCRIPT_DEPLOYMENT	bundleinstallationscriptdeployment
BUSINESS_EVENTS_PROCESSING_HISTORY	businesseventsprocessinghistory
CALENDAR_EVENT	calendarevent
CAMPAIGN_AUDIENCE	campaignaudience
CAMPAIGN_CATEGORY	campaigncategory
CAMPAIGN_CHANNEL	campaignchannel
CAMPAIGN_EMAIL_ADDRESS	campaignemailaddress
CAMPAIGN_EVENT	campaignevent
CAMPAIGN_FAMILY	campaignfamily
CAMPAIGN_OFFER	campaignoffer
CAMPAIGN_RESPONSE	campaignresponse
CAMPAIGN_SEARCH_ENGINE	campaignsearchengine
CAMPAIGN_SUBSCRIPTION	campaignsubscription
CAMPAIGN_TEMPLATE	campaigntemplate
CAMPAIGN_VERTICAL	campaignvertical
CARDHOLDER_AUTHENTICATION	cardholderauthentication
CARDHOLDER_AUTHENTICATION_EVENT	cardholderauthenticationevent
CATEGORY1099MISC	category1099misc
CHARGE	charge
CHARGE_RULE	chargerule
CHARGE_RUN	chargerun
CHARGE_TYPE	chargetype
CLASSIFICATION	classification

Enum Value	Sets Query.type Property To
CLIENT_SCRIPT	clientscript
CLIENT_SCRIPT_DEPLOYMENT	clientscriptdeployment
COMPETITOR	competitor
CONSOLIDATED_EXCHANGE_RATE	consolidatedexchangerate
CONSOLIDATED_RATE_ADJUSTOR_PLUGIN	consolidatedrateadjustorplugin
CONTACT	contact
CONTACT_CATEGORY	contactcategory
CONTACT_ROLE	contactrole
CONTACT_SUBSIDIARY_RELATIONSHIP	contactsubsidiaryrelationship
COST_CATEGORY	costcategory
COUPON_CODE	couponcode
CURRENCY	currency
CURRENCY_RATE	currencyrate
CUSTOMER	customer
CUSTOMER_CATEGORY	customercategory
CUSTOMER_MESSAGE	customermessage
CUSTOMER_SEGMENT	customersegment
CUSTOMER_SUBSIDIARY_RELATIONSHIP	customersubsidiaryrelationship
CUSTOM_FIELD	customfield
CUSTOM_GL_PLUGIN	customglplugin
CUSTOM_LIST	customlist
CUSTOM_RECORD_TYPE	customrecordtype
CUSTOM_SEGMENT	customsegment
CUSTOM_TRANSACTION_TYPE	customtransactiontype
DELETED_RECORD	deletedrecord
DEPARTMENT	department
DEVICE_ID	deviceid
DISTRIBUTION_CATEGORY	distributioncategory
DISTRIBUTION_NETWORK	distributionnetwork
DOMAIN	domain
EMAIL_CAPTURE_PLUGIN	emailcaptureplugin
EMAIL_TEMPLATE	emailtemplate

Enum Value	Sets Query.type Property To
EMPLOYEE	employee
EMPLOYEE_LIST	employeelist
EMPLOYEE_STATUS	employeestatus
EMPLOYEE_SUBSIDIARY_RELATIONSHIP	employeesubsidiaryrelationship
EMPLOYEE_TYPE	employeetype
ENTITY	entity
ENTITY_GROUP	entitygroup
ENTITY_SUBSIDIARY_RELATIONSHIP	entitysubsidiaryrelationship
EXPENSE_CATEGORY	expensecategory
FAX_TEMPLATE	faxtemplate
FILE	file
FISCAL_CALENDAR	fiscalcalendar
FORECAST	forecast
FULFILLMENT_EXCEPTION_REASON	fulfillmentexceptionreason
FULFILLMENT_REQUEST	fulfillmentrequest
GATEWAY_NOTIFICATION	gatewaynotification
GENERAL_ALLOCATION_SCHEDULE	generalallocationschedule
GENERAL_TOKEN	generaltoken
GENERIC_RESOURCE	genericresource
GENERIC_RESOURCE_SUBSIDIARY_RELATIONSHIP	genericresourcesubsidiaryrelationship
GIFT_CERTIFICATE	giftcertificate
GLOBAL_ACCOUNT_MAPPING	globalaccountmapping
GLOBAL_INVENTORY_RELATIONSHIP	globalinventoryrelationship
GL_LINES_AUDIT_LOG	gllinesauditlog
GL_LINES_PLUGIN_REVISION	gllinespluginrevision
G_L_NUMBERING_SEQUENCE	glnumberingsequence
INBOUND_SHIPMENT	inboundshipment
INCOTERM	incoterm
INVENTORY_COST_TEMPLATE	inventorycosttemplate
INVENTORY_NUMBER	inventorynumber
INVENTORY_STATUS	inventorystatus
INV_T_ITEM_PRICE_HISTORY	invitempricehistory

Enum Value	Sets Query.type Property To
ISSUE	issue
ITEM	item
ITEM_ACCOUNT_MAPPING	itemaccountmapping
ITEM_COLLECTION	itemcollection
ITEM_DEMAND_PLAN	itemdemandplan
ITEM_LOCATION_CONFIGURATION	itemlocationconfiguration
ITEM_SEGMENT_INCLUDING_SYNTHETIC	itemsegmentincludingsynthetic
ITEM_SUPPLY_PLAN	itemsupplyplan
I_P_RESTRICTIONS	iprestrictions
JOB	job
JOB_RESOURCE_ROLE	jobresourcerole
JOB_STATUS	jobstatus
JOB_TYPE	jobtype
KNOWLEDGE_BASE	knowledgebase
LOCATION	location
LOCATION_COSTING_GROUP	locationcostinggroup
LOGIN_AUDIT	loginaudit
MAIL_TEMPLATE	mailtemplate
MANUFACTURING_COST_TEMPLATE	manufacturingcosttemplate
MANUFACTURING_ROUTING	manufacturingrouting
MANUFACTURING_TRANSACTION	manufacturingtransaction
MAP_REDUCE_SCRIPT	mapreducescript
MAP_REDUCE_SCRIPT_DEPLOYMENT	mapreducescriptdeployment
MASS_UPDATE_SCRIPT	massupdatescript
MASS_UPDATE_SCRIPT_DEPLOYMENT	massupdatescriptdeployment
MEDIA_ITEM_FOLDER	mediaitemfolder
MEM_DOC	memdoc
MEM_DOC_TRANSACTION_TEMPLATE	memdoctransactiontemplate
MERCHANDISE_HIERARCHY_LEVEL	merchandisehierarchylevel
MERCHANDISE_HIERARCHY_NODE	merchandisehierarchynode
MERCHANDISE_HIERARCHY_VERSION	merchandisehierarchyversion
MESSAGE	message

Enum Value	Sets Query.type Property To
MFG_PLANNED_TIME	mfgplannedtime
NEXUS	nexus
NOTE	note
ONLINE_CASE_FORM	onlinecaseform
ONLINE_FORM_TEMPLATE	onlineformtemplate
ONLINE_LEAD_FORM	onlineleadform
ORDER_ALLOCATION_STRATEGY	orderallocationstrategy
OTHER_NAME	othername
OTHER_NAME_CATEGORY	othernamecategory
OTHER_NAME_SUBSIDIARY_RELATIONSHIP	othernamesubsidiaryrelationship
OUTBOUND_REQUEST	outboundrequest
O_AUTH_TOKEN	oauthtoken
PARTNER	partner
PARTNER_SUBSIDIARY_RELATIONSHIP	partnersubsidiaryrelationship
PAYCHECK	paycheck
PAYMENT_CARD_SEARCH_RECORD	paymentcardsearchrecord
PAYMENT_CARD_TOKEN	paymentcardtoken
PAYMENT_EVENT	paymentevent
PAYMENT_GATEWAY_PLUGIN	paymentgatewayplugin
PAYMENT_INSTRUMENT	paymentinstrument
PAYMENT_METHOD	paymentmethod
PAYMENT_PROCESSING_PROFILE	paymentprocessingprofile
PAYMENT_RESULT_PREVIEW	aymentresultpreview
PAYROLL_BATCH	payrollbatch
PAYROLL_ITEM	payrollitem
PDF_TEMPLATE	pdftemplate
PHONE_CALL	phonecall
PLANNED_STANDARD_COST	plannedstandardcost
PLATFORM_EXTENSION_PLUGIN	platformextensionplugin
PLUG_IN_TYPE	plugintype
PLUG_IN_TYPE_IMPL	plugintypeimpl
PORTLET	portlet

Enum Value	Sets Query.type Property To
PORTLET_DEPLOYMENT	portletdeployment
PREDICTED_RISK_TRAIN_EVAL_HISTORY	predictedrisktrainevalhistory
PRICE_LEVEL	pricelevel
PRICING	pricing
PRICING_GROUP	pricinggroup
PROJECT_SUBSIDIARY_RELATIONSHIP	projectsubsidiaryrelationship
PROJECT_TASK	projecttask
PROJECT_TEMPLATE	projecttemplate
PROJECT_TEMPLATE_SUBSIDIARY_RELATIONSHIP	projecttemplatesubsidiaryrelationship
PROMOTIONS_PLUGIN	promotionsplugin
PROMOTION_CODE	promotioncode
PUBLISHED_SAVED_SEARCH	publishedsavedsearch
QUANTITY_PRICING_SCHEDULE	quantitypricingschedule
QUOTA	quota
RECENT_RECORD	recentrecord
REDIRECT	redirect
RESOURCE_GROUP	resourcegroup
RESTLET	restlet
RESTLET_DEPLOYMENT	restletdeployment
REV_REC_SCHEDULE	revrecschedule
REV_REC_TEMPLATE	revrectemplate
ROLE	role
SALES_INVOICED	salesinvoiced
SALES_ORDERED	salesordered
SALES_TAX_ITEM	salestaxitem
SCHEDULED_SCRIPT	scheduledscript
SCHEDULED_SCRIPT_DEPLOYMENT	scheduledscriptdeployment
SCHEDULED_SCRIPT_INSTANCE	scheduledscriptinstance
SCRIPT	script
SCRIPT_CUSTOM_RECORD_TYPE	scriptcustomrecordtype
SCRIPT_DEPLOYMENT	scriptdeployment
SCRIPT_NOTE	scriptnote

Enum Value	Sets Query.type Property To
SCRIPT_RECORD_TYPE	scriptrecordtype
SEARCH_CAMPAIGN	searchcampaign
SENT_EMAIL	sentemail
SHIPPING_PACKAGE	shippingpackage
SHIPPING_PARTNERS_PLUGIN	shippingpartnersplugin
SHIPPING_PARTNER_REGISTRATION	shippingpartnerregistration
SHIP_ITEM	shipitem
SHOPPING_CART	shoppingcart
SITE_CATEGORY	siterecategory
SOLUTION	solution
STANDARD_COST_VERSION	standardcostversion
STATISTICAL_JOURNAL_ENTRY	statisticaljournalentry
STATISTICAL_SCHEDULE	statisticalschedule
STORE_TAB	storetab
SUBLIST	sublist
SUBSIDIARY	subsidiary
SUBSIDIARY_SETTINGS	subsidiarysettings
SUITELET	suitelet
SUITELET_DEPLOYMENT	suiteletdeployment
SUITE_SCRIPT_DETAIL	suitescriptdetail
SUPPLY_CHAIN_SNAPSHOT	supplychainsnapshot
SUPPLY_CHAIN_SNAPSHOT_SIMULATION	supplychainsnapshotsimulation
SUPPORT_CASE	supportcase
SYSTEM_EMAIL_TEMPLATE	systememailtemplate
SYSTEM_NOTE	systemnote
SYSTEM_NOTE2	systemnote2
SYSTEM_NOTE_FIELD	systemnotefield
TASK	task
TASK_ITEM_STATUS	taskitemstatus
TAX_CALCULATION_PLUGIN	taxcalculationplugin
TAX_TYPE	taxtype
TERM	term

Enum Value	Sets Query.type Property To
TEST_PLUGIN	testplugin
TIME_BILL	timebill
TOPIC	topic
TRANSACTION	transaction
TRANSACTION_DELETION_REASON	transactiondeletionreason
TRANSACTION_HISTORY	transactionhistory
TRANSACTION_NUMBERING_AUDIT_LOG	transactionnumberingauditlog
UMD_FIELD	umdfield
UNDELIVERED_EMAIL	undeliveredemail
UNITS_TYPE	unitstype
USER_EVENT_SCRIPT	userevents
USER_EVENT_SCRIPT_DEPLOYMENT	userevents
USER_O_AUTH_TOKEN	useroauth
USRSAVEDSEARCH	usrsavedsearch
USR_AUDIT_LOG	usrauditlog
USR_EXECUTION_LOG	usrexecutionlog
VENDOR	vendor
VENDOR_CATEGORY	vendorcategory
VENDOR_SUBSIDIARY_RELATIONSHIP	vendorsubsidiaryrelationship
WEBAPP	webapp
WEB_SITE	website
WORKFLOW_ACTION_SCRIPT	workflowactions
WORKFLOW_ACTION_SCRIPT_DEPLOYMENT	workflowactions
WORKPLACE	workplace
WORK_CALENDAR	workcalendar
WBS	wbs

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

```

1 // Add additional code
2 ...

```

```

3  var myCustomerQuery = query.create({
4    type: query.Type.CUSTOMER
5  });
6
7  var mySalesRepJoin = myCustomerQuery.autoJoin({
8    fieldId: 'salesrep'
9  });
10
11 var firstCondition = myCustomerQuery.createCondition({
12   fieldId: 'id',
13   operator: query.Operator.EQUAL,
14   values: 107
15 });
16 var secondCondition = myCustomerQuery.createCondition({
17   fieldId: 'id',
18   operator: query.Operator.EQUAL,
19   values: 2647
20 });
21 var thirdCondition = mySalesRepJoin.createCondition({
22   fieldId: 'email',
23   operator: query.Operator.START_WITH_NOT,
24   values: 'foo'
25 });
26
27 myCustomerQuery.condition = myCustomerQuery.and(
28   thirdCondition, myCustomerQuery.or(firstCondition, secondCondition)
29 );
30
31 var resultSet = myCustomerQuery.run();
32 ...
33 // Add additional code

```

N/record Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the record module to work with NetSuite records. You can use this module to create, delete, copy, load, or make changes to a record.

SuiteScript supports working with standard NetSuite records and with instances of custom record types. Supported standard record types are described in the [SuiteScript Records Browser](#). Refer also to [SuiteScript Supported Records](#). For help working with an instance of a custom record type, see the help topic [Custom Record](#).

For help finding a record's internal ID, see the help topic [How do I find a record's internal ID?](#)



Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). The NetSuite UI should only be accessed with SuiteScript APIs.

- [N/record Module Members](#)
- [Column Object Members](#)
- [Field Object Members](#)
- [Macro Object Members](#)
- [Record Object Members](#)
- [Sublist Object Members](#)
- [N/record Module Script Samples](#)

- N/record Default Values

N/record Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	record.Column	Object	Client and server-side scripts	Encapsulates a column of a sublist on a standard or custom record.
	record.Field	Object	Client and server-side scripts	Encapsulates a body or sublist field on a standard or custom record.
	record.Macro	Object	Client and server-side scripts	Encapsulates a NetSuite record macro.
	Plain JavaScript Object	Object	Client and server-side scripts	A plain JavaScript object of record macros available for a record type. This object is returned by Record.getMacros(options) .
	record.Record	Object	Client and server-side scripts	Encapsulates a NetSuite record.
	record.Sublist	Object	Client and server-side scripts	Encapsulates a sublist on a standard or custom record.
Method	record.attach(options)	void	Client and server-side scripts	Attaches a record to another record.
	record.attach.promise(options)	Promise	Client scripts	Attaches a record asynchronously to another record.
	record.copy(options)	record.Record	Client and server-side scripts	Creates a new record by copying an existing record in NetSuite.
	record.copy.promise(options)	Promise	Client scripts	Creates a new record asynchronously by copying an existing record in NetSuite.
	record.create(options)	record.Record	Client and server-side scripts	Creates a new record.
	record.create.promise(options)	Promise	Client scripts	Creates a new record asynchronously.
	record.delete(options)	number	Client and server-side scripts	Deletes a record.
	record.delete.promise(options)	Promise	Client scripts	Deletes a record asynchronously.
	record.detach(options)	void	Client and server-side scripts	Detaches a record from another record.
	record.detach.promise(options)	Promise	Client scripts	Detaches a record from another record asynchronously.
	record.load(options)	Object	Client and server-side scripts	Loads an existing record.
	record.load.promise(options)	Promise	Client scripts	Loads an existing record asynchronously.
	record.submitFields(options)	Object	Client and server-side scripts	Updates and submits one or more body fields on an existing record in NetSuite, and returns the internal ID of the parent record.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	record.submitFields.promise(options)	Promise	Client scripts	Updates and submits one or more body fields asynchronously on an existing record in NetSuite, and returns the internal ID of the parent record.
	record.transform(options)	record.Record	Client and server-side scripts	Transforms a record from one type into another, using data from an existing record.
	record.transform.promise(options)	Promise	Client scripts	Transforms a record from one type into another asynchronously, using data from an existing record.
Enum	record.Type	enum	Client and server-side scripts	Enumeration that holds the string values for supported standard record types.

Column Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	Column.id	string (read-only)	Client and server-side scripts	Returns the internal ID of the column.
	Column.isDisabled	boolean true false	Client and server-side scripts	Indicates whether the column is disabled.
	Column.isDisplay	boolean true false	Client and server-side scripts	Indicates whether the column is displayed.
	Column.isMandatory	boolean true false	Client and server-side scripts	Indicates whether the column is mandatory.
	Column.isSortable	boolean true false (read-only)	Client and server-side scripts	Indicates whether the column is sortable.
	Column.label	string (read-only)	Client and server-side scripts	Returns the UI label for the column.
	Column.sublistId	string (read-only)	Client and server-side scripts	Returns the internal ID of the standard or custom sublist that contains the column.
	Column.type	string (read-only)	Client and server-side scripts	Returns the column type.

Field Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Field.getSelectOptions(options)	array	Client and server-side scripts	Returns an array of available options on a standard or custom select, multiselect, or radio field as key-value pairs. Only the first 1,000 available options are returned.
Property	Field.label	string (read-only)	Client and server-side scripts	Returns the UI label for a standard or custom field body or sublist field.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Field.id	string (read-only)	Client and server-side scripts	Returns the internal ID of a standard or custom body or sublist field.
	Field.type	string (read-only)	Client and server-side scripts	Returns the type of a body or sublist field.
	Field.isMandatory	boolean true false	Client and server-side scripts	Returns true if the standard or custom field is mandatory on the record form, or false otherwise.
	Field.sublistId	string (read-only)	Client and server-side scripts	Returns the ID of the sublist associated with the specified sublist field.
	Field.isDisplay	boolean true false	Client and server-side scripts	Returns true if the field is visible on the record form, or false if it is not.

Macro Object Members

The following members are called on the `record.Macro` object. For information about record macros, see the help topic [Overview of Record Action and Macro APIs](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Macro.execute(options)	Object	Client and server-side scripts	Performs a macro operation and returns its result in an object.
	Macro.execute.promise(options)	Promise	Client scripts	Performs a macro operation asynchronously.
	Macro(options)	Object	Client and server-side scripts	Performs a macro operation and returns its result in an object.
	Macro.promise(options)	Promise	Client scripts	Performs a macro operation asynchronously.
Property	Macro.id	string	Client and server-side scripts	The ID of the macro. For a list of macro IDs, see the help topic Supported Record Macros
	Macro.label	string	Client and server-side scripts	The macro label.
	Macro.description	string	Client and server-side scripts	The macro description.
	Macro.attributes	Object	Client and server-side scripts	The macro defined attributes.

Record Object Members

The following members are called on the `record.Record` object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Record.cancelLine(options)	record.Record	Client and server-side scripts	Cancels the currently selected line on a sublist.
	Record.commitLine(options)	record.Record	Client and server-side scripts	Commits the currently selected line on a sublist.
	Record.executeMacro(options)	Object	Client and server-side scripts	Performs macro operation and returns its result in a plain JavaScript object.
	Record.getMacros(options)	Object	Client and server-side scripts	Provides a plain JavaScript object that contains macro objects defined for a record type, indexed by the Macro ID.
	Record.findMatrixSublistLineWithValue(options)	number	Client and server-side scripts	Returns the line number of the first instance where a specified value is found in a specified column of the matrix.
	Record.findSublistLineWithValue(options)	number	Client and server-side scripts	Returns the line number for the first occurrence of a field value in a sublist.
	Record.getCurrentMatrixSublistValue(options)	number Date string array boolean true false	Client and server-side scripts	Gets the value for the currently selected line in the matrix.
	Record.getCurrentSublistField(options)	record.Field	Client and server-side scripts	Returns a field object from a sublist.
	Record.getCurrentSublistIndex(options)	number	Client and server-side scripts	Returns the line number of the currently selected line.
	Record.getCurrentSublistSubrecord(options)	record.Record	Client and server-side scripts	Gets the subrecord for the associated sublist field on the current line.
	Record.getCurrentSublistText(options)	string	Client and server-side scripts	Returns a text representation of the field value in the currently selected line.
	Record.getCurrentSublistValue(options)	number Date string array boolean true false	Client and server-side scripts	Returns the value of a sublist field on the currently selected sublist line.
	Record.getField(options)	record.Field	Client and server-side scripts	Returns a field object from a record.
	Record.getFields()	string[]	Client and server-side scripts	Returns the body field names (internal ids) of all the fields in the record, including machine header field and matrix header fields.
	Record.getLineCount(options)	number	Client and server-side scripts	Returns the number of lines in a sublist.
	Record.getMacro(options)	record.Macro	Client and server-side scripts	Provides a macro to execute.
	Record.getMacros(options)	Object	Client and server-side scripts	Provides a plain JavaScript object that contains macro objects defined for

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				a record type, indexed by the Macro ID.
	Record.getMatrixHeaderCount(options)	number	Client and server-side scripts	Returns the number of columns for the specified matrix.
	Record.getMatrixHeaderField(options)	record.Field	Client and server-side scripts	Gets the field for the specified header in the matrix.
	Record.getMatrixHeaderValue(options)	number Date string array boolean true false	Client and server-side scripts	Gets the value for the associated header in the matrix.
	Record.getMatrixSublistField(options)	record.Field	Client and server-side scripts	Gets the field for the specified sublist in the matrix.
	Record.getMatrixSublistValue(options)	number Date string array boolean true false	Client and server-side scripts	Gets the value for the associated field in the matrix.
	Record.getSublist(options)	record.Sublist	Client and server-side scripts	Returns the specified sublist.
	Record.getSublists()	string[]	Client and server-side scripts	Returns all the names of all the sublists.
	Record.getSublistField(options)	record.Field	Client and server-side scripts	Returns a field object from a sublist.
	Record.getSublistFields(options)	string[]	Client and server-side scripts	Returns all the field names in a sublist.
	Record.getSublistSubrecord(options)	record.Record	Client and server-side scripts	Gets the subrecord associated with a sublist field. (standard mode only)
	Record.getSublistText(options)	string	Client and server-side scripts	Returns the value of a sublist field in a text representation.
	Record.getSublistValue(options)	number Date string array boolean true false	Client and server-side scripts	Returns the value of a sublist field.
	Record.getSubrecord(options)	record.Record	Client and server-side scripts	Gets the subrecord for the associated field.
	Record.getText(options)	string	Client and server-side scripts	Returns the text representation of a field value.
	Record.getValue(options)	number Date string array boolean true false	Client and server-side scripts	Returns the value of a field.
	Record.hasCurrentSublistSubrecord(options)	boolean true false	Client and server-side scripts	Returns a value indicating whether the associated sublist field has a subrecord on the current line.
	Record.hasSublistSubrecord(options)	boolean true false	Client and server-side scripts	Returns a value indicating whether the associated sublist field contains a subrecord.
	Record.hasSubrecord(options)	boolean true false	Client and server-side scripts	Returns a value indicating whether the field contains a subrecord.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Record.insertLine(options)	record.Record	Client and server-side scripts	Inserts a sublist line.
	Record.removeCurrentSublistSubrecord(options)	record.Record	Client and server-side scripts	Removes the subrecord for the associated sublist field on the current line.
	Record.removeLine(options)	record.Record	Client and server-side scripts	Removes a sublist line.
	Record.removeSublistSubrecord(options)	record.Record	Client and server-side scripts	Removes the subrecord for the associated sublist field. (standard mode only)
	Record.removeSubrecord(options)	record.Record	Client and server-side scripts	Removes the subrecord for the associated field.
	Record.save(options)	number	Client and server-side scripts	Submits a new record or saves edits to an existing record. This method is not available to subrecords.
	Record.save.promise(options)	number	Client scripts	Submits a new record asynchronously or saves edits to an existing record asynchronously. This method is not available to subrecords.
	Record.selectLine(options)	record.Record	Client and server-side scripts	Selects an existing line in a sublist.
	Record.selectNewLine(options)	record.Record	Client and server-side scripts	Selects a new line at the end of a sublist.
	Record.setCurrentMatrixSublistValue(options)	record.Record	Client and server-side scripts	Sets the value for the line currently selected in the matrix.
	Record.setCurrentSublistText(options)	record.Record	Client and server-side scripts	Sets the value for the field in the currently selected line by a text representation.
	Record.setCurrentSublistValue(options)	record.Record	Client and server-side scripts	Sets the value for the field in the currently selected line.
	Record.setMatrixHeaderValue(options)	record.Record	Client and server-side scripts	Sets the value for the associated header in the matrix.
	Record.setMatrixSublistValue(options)	record.Record	Client and server-side scripts	Sets the value for the associated field in the matrix.
	Record.setSublistText(options)	record.Record	Client and server-side scripts	Sets the value of a sublist field by a text representation. (standard mode only)
	Record.setSublistValue(options)	record.Record	Client and server-side scripts	Sets the value of a sublist field. (standard mode only)
	Record.setText(options)	record.Record	Client and server-side scripts	Sets the value of the field by a text representation.
	Record.setValue(options)	record.Record	Client and server-side scripts	Sets the value of a field.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	Record.id	number (read-only)	Client and server-side scripts	The internal ID of a specific record. This property is not available to subrecords.
	Record.isDynamic	boolean (read-only)	Client and server-side scripts	Indicates whether the record is in dynamic or standard mode.
	Record.type	string (read-only)	Client and server-side scripts	The record type. This property is not available to subrecords.

Sublist Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Sublist.getColumn(options)	record.Column	Client and server-side scripts	Returns a column in the sublist.
Property	Sublist.id	string (read-only)	Client and server-side scripts	Returns the internal ID of the sublist.
	Sublist.isChanged	boolean true false (read-only)	Client and server-side scripts	Indicates whether the sublist has changed on the record form.
	Sublist.isDisplay	boolean true false (read-only)	Client and server-side scripts	Indicates whether the sublist is displayed on the record form.
	Sublist.type	string (read-only)	Client and server-side scripts	Returns the sublist type.

N/record Module Script Samples

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to use the record module to create and save a Contact record..

Important: Some of the values in these samples are placeholders. Before using these samples, replace all hardcoded values, such as IDs and file paths, with valid values from your NetSuite account. If you run a script with an invalid value, the system may throw an error.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/record'], function(record) {
6     function createAndSaveContactRecord() {
7         var nameData = {
8             firstname: 'John',
9             middlename: 'Doe',
10            lastname: 'Smith'
11        };

```

```

12     var objRecord = record.create({
13         type: record.Type.CONTACT,
14         isDynamic: true
15     });
16     objRecord.setValue({
17         fieldId: 'subsidiary',
18         value: '1'
19     });
20     for ( var key in nameData) {
21         if (nameData.hasOwnProperty(key)) {
22             objRecord.setValue({
23                 fieldId: key,
24                 value: nameData[key]
25             });
26         }
27     }
28     var recordId = objRecord.save({
29         enableSourcing: false,
30         ignoreMandatoryFields: false
31     });
32 }
33 createAndSaveContactRecord();
34 });

```

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to create and save a contact record using promise methods.

i Note: To debug client scripts like the following, we recommend that you use Chrome DevTools for Chrome, Firebug debugger for Firefox, or Microsoft Script Debugger for Internet Explorer. For information about these tools, see the documentation provided with each browser. For more information on debugging SuiteScript client scripts, see the help topic [Debugging Client Scripts](#).

```

1 /**
2 * @NApiVersion 2.x
3 */
4
5 require(['N/record'], function(record) {
6     function createAndSaveContactRecordWithPromise() {
7         var nameData = {
8             firstname: 'John',
9             middlename: 'Doe',
10            lastname: 'Smith'
11        };
12        var createRecordPromise = record.create.promise({
13            type: record.Type.CONTACT,
14            isDynamic: true
15        });
16
17        createRecordPromise.then(function(objRecord) {
18            console.log('start evaluating promise content');
19            objRecord.setValue({
20                fieldId: 'subsidiary',
21                value: '1'
22            });
23            for ( var key in nameData) {
24                if (nameData.hasOwnProperty(key)) {
25                    objRecord.setValue({
26                        fieldId: key,
27                        value: nameData[key]
28                    });
29                }
30            }
31            var recordId = objRecord.save({
32                enableSourcing: false,

```

```

33     ignoreMandatoryFields: false
34   });
35   }, function(e) {
36     log.error('Unable to create contact', e.name);
37   });
38 }
39 createAndSaveContactRecordWithPromise();
40 });

```



Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to access sublists and a subrecord from a record. This sample requires the Advanced Number Inventory Management feature.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/record'], function(record) {
6   function createPurchaseOrder() {
7     var rec = record.create({
8       type: 'purchaseorder',
9       isDynamic: true
10    });
11    rec.setValue({
12      fieldId: 'entity',
13      value: 52
14    });
15    rec.setValue({
16      fieldId: 'location',
17      value: 2
18    });
19    rec.selectNewLine({
20      sublistId: 'item'
21    });
22    rec.setCurrentSublistValue({
23      sublistId: 'item',
24      fieldId: 'item',
25      value: 190
26    });
27    rec.setCurrentSublistValue({
28      sublistId: 'item',
29      fieldId: 'quantity',
30      value: 2
31    });
32    subrecordInvDetail = rec.getCurrentSublistSubrecord({
33      sublistId: 'item',
34      fieldId: 'inventorydetail'
35    });
36    subrecordInvDetail.selectNewLine({
37      sublistId: 'inventoryassignment'
38    });
39    subrecordInvDetail.setCurrentSublistValue({
40      sublistId: 'inventoryassignment',
41      fieldId: 'receiptinventorynumber',
42      value: 'myinventoryNumber'
43    });
44    subrecordInvDetail.commitLine({
45      sublistId: 'inventoryassignment'
46    });
47    subrecordInvDetail.selectLine({
48      sublistId: 'inventoryassignment',
49      line: 0
50    });
51    var myInventoryNumber = subrecordInvDetail.getCurrentSublistValue({
52      sublistId: 'inventoryassignment',
53      fieldId: 'receiptinventorynumber'
54    });

```

```

54     });
55     rec.commitLine({
56       sublistId: 'item'
57     });
58     var recordId = rec.save();
59   }
60   createPurchaseOrder();
61 });

```

Note: For additional script samples that include subrecords, see the help topic [SuiteScript 2.0 Scripting Subrecords](#).

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to access sublists and a subrecord from a record using promise methods. This sample requires the Advanced Number Inventory Management feature.

Note: To debug client scripts like the following, we recommend that you use Chrome DevTools for Chrome, Firebug debugger for Firefox, or Microsoft Script Debugger for Internet Explorer. For information about these tools, see the documentation provided with each browser. For more information on debugging SuiteScript client scripts, see the help topic [Debugging Client Scripts](#).

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/record'], function(record) {
6   function createPurchaseOrder() {
7     var createRecordPromise = record.create.promise({
8       type: 'purchaseorder',
9       isDynamic: true
10    });
11    createRecordPromise.then(function(rec) {
12      rec.setValue({
13        fieldId: 'entity',
14        value: 52
15      });
16      rec.setValue({
17        fieldId: 'location',
18        value: 2
19      });
20      rec.selectNewLine({
21        sublistId: 'item'
22      });
23      rec.setCurrentSublistValue({
24        sublistId: 'item',
25        fieldId: 'item',
26        value: 190
27      });
28      rec.setCurrentSublistValue({
29        sublistId: 'item',
30        fieldId: 'quantity',
31        value: 2
32      });
33      subrecordInvDetail = rec.getCurrentSublistSubrecord({
34        sublistId: 'item',
35        fieldId: 'inventorydetail'
36      });
37      subrecordInvDetail.selectNewLine({
38        sublistId: 'inventoryassignment'
39      });
40      subrecordInvDetail.setCurrentSublistValue({
41        sublistId: 'inventoryassignment',
42        fieldId: 'receiptinventorynumber',

```

```

43     value: 'myinventoryNumber'
44   });
45   subrecordInvDetail.commitLine({
46     sublistId: 'inventoryassignment'
47   });
48   subrecordInvDetail.selectLine({
49     sublistId: 'inventoryassignment',
50     line: 0
51   });
52   var myInventoryNumber = subrecordInvDetail.getCurrentSublistValue({
53     sublistId: 'inventoryassignment',
54     fieldId: 'receiptinventorynumber'
55   });
56   rec.commitLine({
57     sublistId: 'item'
58   });
59   var recordId = rec.save();
60 }, function(err) {
61   log.error('Unable to create purchase order!', err.name);
62 });
63 }
64 createPurchaseOrder();
65 });

```

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows you how to call a calculateTax macro on a sales order record. To execute a macro on a record, the record must be created or loaded in dynamic mode. Note that the SuiteTax feature must be enabled to successfully execute the macro used in this sample.

For information about record macros, see the help topic [Overview of Record Action and Macro APIs](#).

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/record'],
6   function(record) {
7
8   var recordObj = record.create({
9     type: record.Type.SALES_ORDER,
10    isDynamic: true
11  });
12
13  var ENTITY_VALUE = 1;
14  var ITEM_VALUE = 1;
15  recordObj.setValue({
16    fieldId: 'entity',
17    value: ENTITY_VALUE
18  });
19  recordObj.selectNewLine({
20    sublistId: 'item'
21  });
22  recordObj.setCurrentSublistValue({
23    sublistId: 'item',
24    fieldId: 'item',
25    value: ITEM_VALUE
26  });
27  recordObj.setCurrentSublistValue({
28    sublistId: 'item',
29    fieldId: 'quantity',
30    value: 1
31  });
32  recordObj.commitLine({
33    sublistId: 'item'
34 });

```

```

36 |     var totalBeforeTax = recordObj.getValue({fieldId: 'total'});
37 |
38 |     // get macros available on the record
39 |     var macros = recordObj.getMacros();
40 |
41 |     // execute the macro
42 |     if ('calculateTax' in macros)
43 |     {
44 |         macros.calculateTax(); // For promise version use: macros.calculateTax.promise()
45 |     }
46 |     // Alternative (direct) macro execution
47 |     // var calculateTax = recordObj.getMacro({id: 'calculateTax'});
48 |     // calculateTax(); // For promise version use: calculateTax.promise()
49 |     var totalAfterTax = recordObj.getValue({fieldId: 'total'});
50 |
51 |     var recordId = recordObj.save({
52 |         enableSourcing: false,
53 |         ignoreMandatoryFields: false
54 |     });
55 | });

```

N/record Default Values

You can use SuiteScript 2.0 to specify record initialization parameters that default when creating, copying, loading, and transforming records. To enable this behavior, use the optional `defaultValues` parameter in the following APIs:

- [record.create\(options\)](#)
- [record.copy\(options\)](#)
- [record.transform\(options\)](#)
- [record.load\(options\)](#)

The following table lists initialization types that are available to certain SuiteScript-supported records and the values they can contain.

Record	Initialization Type	Values
All SuiteScript-supported records that support form customization.	customform	<customformid>
Assembly Build	assemblyitem	<assemblyitemid>
Cash Refund	entity	<entityid>
Cash Sale	entity	<entityid>
Charge Rule	entity	<entityid>
Check	entity	<entityid>
Credit Memo	entity	<entityid>
Customer Payment	entity	<entityid>
Customer Refund	entity	<entityid>
Deposit	disablepaymentfilters	<disablepaymentfilters>
Estimate	entity	<entityid>
Expense Report	entity	<entityid>
Invoice	entity	<entityid>
Item Receipt	entity	<entityid>

Record	Initialization Type	Values
Non-Inventory Part	subtype	sale resale purchase
Opportunity	entity	<entityid>
Other Charge Item	subtype	sale resale purchase
Purchase Order	entity	<entityid>
Return Authorization	entity	<entityid>
Sales Order	entity	<entityid>
Script Deployment	script	<scriptid>
Service	subtype	sale resale purchase
Subscription Change Order	entity	<scriptid>
Tax Group	nexuscountry	<countrycode> See Country Codes Used for Initialization Parameters .
Tax Type	country	<countrycode> See Country Codes Used for Initialization Parameters .
Topic	parenttopic	<parenttopicid>
Vendor Bill	entity	<entityid>
Vendor Payment	entity	<entityid>
Work Order	assemblyitem	<assemblyitemid>

Country Codes Used for Initialization Parameters

If you are scripting the Tax Group or Tax Type records, you can initialize the record to source all values related to a specific country. In your script, use the country code for the *countrycodeid* value, for example:

```
1 | record.create('taxgroup', {nexuscountry: 'AR'});
```

Country Code	Country Name
AD	Andorra
AE	United Arab Emirates
AF	Afghanistan
AG	Antigua and Barbuda
AI	Anguilla
AL	Albania
AM	Armenia
AO	Angola

Country Code	Country Name
AQ	Antarctica
AR	Argentina
AS	American Samoa
AT	Austria
AU	Australia
AW	Aruba
AX	Aland Islands
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BD	Bangladesh
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BH	Bahrain
BI	Burundi
BJ	Benin
BL	Saint Barthélemy
BM	Bermuda
BN	Brunei Darrussalam
BO	Bolivia
BQ	Bonaire, Saint Eustatius, and Saba
BR	Brazil
BS	Bahamas
BT	Bhutan
BV	Bouvet Island
BW	Botswana
BY	Belarus
BZ	Belize
CA	Canada
CC	Cocos (Keeling) Islands
CD	Congo, Democratic People's Republic

Country Code	Country Name
CF	Central African Republic
CG	Congo, Republic of
CH	Switzerland
CI	Cote d'Ivoire
CK	Cook Islands
CL	Chile
CM	Cameroon
CN	China
CO	Colombia
CR	Costa Rica
CU	Cuba
CV	Cape Verde
CW	Curacao
CX	Christmas Island
CY	Cyprus
CZ	Czech Republic
DE	Germany
DJ	Djibouti
DK	Denmark
DM	Dominica
DO	Dominican Republic
DZ	Algeria
EA	Ceuta and Melilla
EC	Ecuador
EE	Estonia
EG	Egypt
EH	Western Sahara
ER	Eritrea
ES	Spain
ET	Ethiopia
FI	Finland
FJ	Fiji

Country Code	Country Name
FK	Falkland Islands
FM	Micronesia, Federal State of
FO	Faroe Islands
FR	France
GA	Gabon
GB	United Kingdom
GD	Grenada
GE	Georgia
GF	French Guiana
GG	Guernsey
GH	Ghana
GI	Gibraltar
GL	Greenland
GM	Gambia
GN	Guinea
GP	Guadeloupe
GQ	Equatorial Guinea
GR	Greece
GS	South Georgia
GT	Guatemala
GU	Guam
GW	Guinea-Bissau
GY	Guyana
HK	Hong Kong
HM	Heard and McDonald Islands
HN	Honduras
HR	Croatia/Hrvatska
HT	Haiti
HU	Hungary
IC	Canary Islands
ID	Indonesia
IE	Ireland

Country Code	Country Name
IL	Israel
IM	Isle of Man
IN	India
IO	British Indian Ocean Territory
IQ	Iraq
IR	Iran (Islamic Republic of)
IS	Iceland
IT	Italy
JE	Jersey
JM	Jamaica
JO	Jordan
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KH	Cambodia
KI	Kiribati
KM	Comoros
KN	Saint Kitts and Nevis
KP	Korea, Democratic People's Republic
KR	Korea, Republic of
KW	Kuwait
KY	Cayman Islands
KZ	Kazakhstan
LA	Lao People's Democratic Republic
LB	Lebanon
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg

Country Code	Country Name
LV	Latvia
LY	Libya
MA	Morocco
MC	Monaco
MD	Moldova, Republic of
ME	Montenegro
MF	Saint Martin
MG	Madagascar
MH	Marshall Islands
MK	Macedonia
ML	Mali
MM	Myanmar
MN	Mongolia
MO	Macau
MP	Northern Mariana Islands
MQ	Martinique
MR	Mauritania
MS	Montserrat
MT	Malta
MU	Mauritius
MV	Maldives
MW	Malawi
MX	Mexico
MY	Malaysia
MZ	Mozambique
NA	Namibia
NC	New Caledonia
NE	Niger
NF	Norfolk Island
NG	Nigeria
NI	Nicaragua
NL	Netherlands

Country Code	Country Name
NO	Norway
NP	Nepal
NR	Nauru
NU	Niue
NZ	New Zealand
OM	Oman
PA	Panama
PE	Peru
PF	French Polynesia
PG	Papua New Guinea
PH	Philippines
PK	Pakistan
PL	Poland
PM	St. Pierre and Miquelon
PN	Pitcairn Island
PR	Puerto Rico
PS	State of Palestine
PT	Portugal
PW	Palau
PY	Paraguay
QA	Qatar
RE	Reunion Island
RO	Romania
RS	Serbia
RU	Russian Federation
RW	Rwanda
SA	Saudi Arabia
SB	Solomon Islands
SC	Seychelles
SD	Sudan
SE	Sweden
SG	Singapore

Country Code	Country Name
SH	Saint Helena
SI	Slovenia
SJ	Svalbard and Jan Mayen Islands
SK	Slovak Republic
SL	Sierra Leone
SM	San Marino
SN	Senegal
SO	Somalia
SR	Suriname
SS	South Sudan
ST	Sao Tome and Principe
SV	El Salvador
SX	Sint Maarten
SY	Syrian Arab Republic
SZ	Swaziland
TC	Turks and Caicos Islands
TD	Chad
TF	French Southern Territories
TG	Togo
TH	Thailand
TJ	Tajikistan
TK	Tokelau
TM	Turkmenistan
TN	Tunisia
TO	Tonga
TP	East Timor
TR	Turkey
TT	Trinidad and Tobago
TV	Tuvalu
TW	Taiwan
TZ	Tanzania
UA	Ukraine

Country Code	Country Name
UG	Uganda
UM	US Minor Outlying Islands
US	United States
UY	Uruguay
UZ	Uzbekistan
VA	Holy See (City Vatican State)
VC	Saint Vincent and the Grenadines
VE	Venezuela
VG	Virgin Islands (British)
VI	Virgin Islands (USA)
VN	Vietnam
VU	Vanuatu
WF	Wallis and Futuna Islands
WS	Samoa
XK	Kosovo
YE	Yemen
YT	Mayotte
ZA	South Africa
ZM	Zambia
ZW	Zimbabwe

record.Column

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a column of a sublist on a standard or custom record. For a complete list of this object's properties, see Column Object Members . This object does not return a value, it returns information about the sublist column.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 var objColumn = objSublist.getColumn({
13   fieldId: 'item'
14 });
15
16 if(objColumn.label === 'myLabel'){
17   //Perform an action
18 }
19 if(objColumn.type === 'checkbox'){
20   //Perform an action
21 }
22 ...
23 // Add additional code.

```

Column.id

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of the column. Note that the Column.id value is the same as the value that is passed into fieldID.
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Column Object Members
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'

```

```

10 });
11
12 var objColumn = objSublist.getColumn({
13   fieldId: 'item'
14 });
15 log.debug ({
16   title: 'ID comparison',
17   details: 'Note that objColumn.id = '+ objColumn.id + ' is the same as the value you passed in as fieldID.'
18 })...
19 // Add additional code.

```

Column.isDisabled

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the column is disabled.
Type	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Column Object Members
Since	2020.2

Syntax

! **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistObj = recordObj.getSublist({
4   sublistId: 'sublistId'
5 });
6 var columnObj = sublistObj.getColumn({
7   fieldId: 'columnId'
8 });
9 //set
10 columnObj.isDisabled = true;
11 //get
12 var isDisabled = columnObj.isDisabled;
13 })...
14 // Add additional code.

```

Column.isDisplay

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the column is displayed.
Type	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/record Module
Sibling Object Members	Column Object Members
Since	2020.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistObj = recordObj.getSublist({
4     sublistId: 'sublistId'
5 });
6 var columnObj = sublistObj.getColumn({
7     fieldId: 'columnId'
8 });
9
10 //set
11 columnObj.isDisplay = false;
12 //get
13 var isDisplay = columnObj.isDisplay;
14 })...
15 // Add additional code.

```

Column.isMandatory

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the column is mandatory.
Type	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Column Object Members
Since	2020.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistObj = recordObj.getSublist({
4     sublistId: 'sublistId'
5 });
6 var columnObj = sublistObj.getColumn({
7     fieldId: 'columnId'
8 });
9 //set

```

```

10 | columnObj.isMandatory = true;
11 | //get
12 | var isMandatory = columnObj.isMandatory;
13 | })...
14 | // Add additional code.

```

Column.isSortable



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the column is sortable.
Type	boolean true false (read-only)
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Column Object Members
Since	2020.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3
4 var sublistObj = recordObj.getSublist({
5     sublistId: 'sublistId'
6 });
7 var columnObj = sublistObj.getColumn({
8     fieldId: 'columnId'
9 });
10 //get
11 var isSortable = columnObj.isSortable;
12 })...
13 // Add additional code.

```

Column.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of the column. This property does not return a value, it returns information about the column label.
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module

Sibling Object Members	Column Object Members
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 var objColumn = objSublist.getColumn({
13   fieldId: 'item'
14 });
15
16 if(objColumn.label === 'myLabel'){
17   //Perform an action
18 }
19 ...
20 // Add additional code.
```

Column.sublistId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of the standard or custom sublist that contains the column.
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Column Object Members
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({}
```

```

9    sublistId: 'item'
10   });
11
12  var objColumn = objSublist.getColumn({
13    fieldId: 'item'
14  });
15 //Perform an action with the objColumn.sublistId value
16 ...
17 // Add additional code.

```

Column.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the column type. For more information on possible return values, see format.Type .
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Column Object Members
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10});
11
12 var objColumn = objSublist.getColumn({
13   fieldId: 'item'
14 });
15
16 if(objColumn.type === 'checkbox'){
17   //Perform an action
18 }
19 ...
20 // Add additional code.

```

record.Field



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a body or sublist field on a standard or custom record.
---------------------------	--

	<p>Use the following methods to access the Field object:</p> <ul style="list-style-type: none"> ▪ Record.getField(options) ▪ Record.getSublistField(options) ▪ Record.getCurrentSublistField(options) ▪ CurrentRecord.getField(options) ▪ CurrentRecord.getSublistField(options) <p>For a complete list of this object's methods and properties, see Field Object Members.</p>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 var objField = objSublist.getField({
13   fieldId: 'item'
14 });
15 if(objField.label === 'myLabel'){
16   //Perform an action
17 }
18 if(objField.type === 'checkbox'){
19   //Perform an action
20 }
21 ...
22 // Add additional code.

```

Field.getSelectOptions(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an array of available options on a standard or custom select, multi-select, or radio field as key-value pairs.
Returns	array



Important: You can only use this method on a record in dynamic mode. For additional information on dynamic mode, see [record.Record](#) and [SuiteScript 2.0 – Standard and Dynamic Modes](#).

	<p>Only the first 1,000 available options are returned in an array.</p> <p>If there are more than 1,000 available options, an empty array [] is returned.</p> <p>This function returns an array in the following format:</p> <pre>1 [{value: 5, text: 'abc'}, {value: 6, text: '123'}]</pre> <p>This function returns Type <code>Error</code> if the field is not a supported field for this method.</p>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Field Object Members
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
<code>options.filter</code>	string	Required	<p>The search string to filter the select options that are returned.</p> <p> Note: Filter values are case insensitive.</p>	2015.2
<code>options.operator</code>	string	Required	<p>The following operators are supported:</p> <ul style="list-style-type: none"> ■ <code>contains</code> (default) ■ <code>is</code> ■ <code>startswith</code> 	2015.2

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
---	--

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 var options = objField.getSelectOptions({
13   filter : 'C',
14   operator : 'startswith'
15 });

```

```

16 //Perform an action with the options array
17 ...
18 // Add additional code.
19

```

Field.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the UI label for a standard or custom field body or sublist field.
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Field Object Members
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 var objField = objSublist.getField({
13   fieldId: 'item'
14 });
15
16 if(objField.label === 'myLabel') {
17   //Perform an action
18 }
19 ...
20 // Add additional code.

```

Field.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of a standard or custom body or sublist field.
Type	string (read-only)
Supported Script Types	Client and server-side scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Field Object Members
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4     type: record.Type.SALES_ORDER,
5     id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9     sublistId: 'item'
10 });
11
12 var objField = objSublist.getField({
13     fieldId: 'item'
14 });
15 //Perform an action with the objField.id value
16 ...
17 //Add additional code.

```

Field.type

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the type of a body or sublist field. For example, the value can return text, date, currency, select, checkbox, etc. For more information on possible return values, see format.Type . The maximum character limit for select field types is 801.
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Field Object Members
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```
1 // Add additional code.
```

```

2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 var objField = objSublist.getField({
13   fieldId: 'item'
14 });
15
16 if(objField.type === 'checkbox'){
17   //Perform an action
18 }
19 ...
20 // Add additional code.

```

Field.isMandatory

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns true if the standard or custom field is mandatory on the record form, or false otherwise.
Type	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Field Object Members
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 var objField = objSublist.getField({
13   fieldId: 'item'
14 });
15
16 if(objField.isMandatory){
17   var options = {
18     title:'Incomplete Field:',
19     message: 'Please complete this field.'
20   };

```

```

21 dialog.alert(options);
22 ...
23 // Add additional code.

```

Field.sublistId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the sublist ID for the specified sublist field.
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Field Object Members
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 var objField = objSublist.getField({
13   fieldId: 'item'
14 });
15
16 //Perform an action with the objField.sublistId
17 ...
18 // Add additional code.

```

Field.isDisplay



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns true if the field is visible on the record form, or false if it is not.
Type	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module

Sibling Object Members	Field Object Members
Since	2015.2

Syntax

```

1 // Add additional code.
2
3 define(['N/currentRecord'], function(currentRecord)
4 {return {pageInit: function(context) {
5     var record = currentRecord.get();
6     var field = record.getField({fieldId:"custbody1"});
7     field.isDisplay = false;
8 ...
9 // Add additional code.

```

record.Macro



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a record macro. For information about record macros, see the help topic Overview of Record Action and Macro APIs . Use the Record.getMacro(options) method to access the Macro object. For a complete list of this object's methods and properties, see Macro Object Members .
Supported Script Types	Client and server-side scripts. For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Methods and Properties	Macro Object Members
Since	2018.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myMacro = record.getMacro({id: 'calculateTax'})
4 ...
5 // Add additional code

```

Macro.execute(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs a macro operation and returns its result in a plain JavaScript object.
---------------------------	---

	For information about record macros, see the help topic Overview of Record Action and Macro APIs .
Returns	{notifications: [], response: {}}
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Parent Object	record.Macro
Sibling Object Members	Macro Object Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.params	Object	optional	The macro arguments.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 timesheet.executeMacro({id:'copyFromWeek', params: {weekOf : '7/10/2017', copyExact : true}});
4 ...
5 // Add additional code

```

Macro.execute.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs a macro operation asynchronously. For information about record macros, see the help topic Overview of Record Action and Macro APIs .
>Returns	Promise Object
Supported Script Types	Client-side scripts

	For additional information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/record Module
Parent Object	record.Macro
Sibling Object Members	Macro Object Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.params	Object	optional	The macro arguments.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 myMacro.execute.promise().then(function(result){ /* do something with macro result */ });
4 ...
5 // Add additional code

```

Macro(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs a macro operation and returns its result in a plain JavaScript object.  Note: Substitute Macro with the name of the macro you are executing. For information about record macros, see the help topic Overview of Record Action and Macro APIs .
Returns	{notifications: [], response: {}}
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Parent Object	record.Macro
Sibling Object Members	Macro Object Members

Since	2018.2
-------	--------

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.params	Object	optional	The macro arguments.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var calculateTax = recordObj.getMacro({id: 'calculateTax'});
4 calculateTax();
5 ...
6 // Add additional code

```

Macro.promise(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs a macro operation asynchronously. i Note: Substitute Macro with the name of the macro you are executing. For information about record macros, see the help topic Overview of Record Action and Macro APIs . i Note: The parameters and errors thrown for this method are the same as those for Macro(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client-side scripts For additional information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/record Module
Parent Object	record.Macro
Sibling Object Members	Macro Object Members
Since	2018.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.params	Object	optional	The macro arguments.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 var calculateTax = recordObj.getMacro({id: 'calculateTax'});
4 calculateTax.promise();
5 ...
6 // Add additional code

```

Macro.id

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the macro. For information about record macros, see the help topic Overview of Record Action and Macro APIs .
Type	string
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Parent Object	record.Macro
Sibling Object Members	Macro Object Members
Since	2018.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var id = macro.id; // get the id of the macro
4 ...
5 // Add additional code

```

Macro.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The label of the macro. For information about record macros, see the help topic Overview of Record Action and Macro APIs .
Type	string
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Parent Object	record.Macro
Sibling Object Members	Macro Object Members
Since	2018.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var label = macro.label; // get the label of the macro
4 ...
5 // Add additional code

```

Macro.description



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The description of the macro. For information about record macros, see the help topic Overview of Record Action and Macro APIs .
Type	string
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Parent Object	record.Macro
Sibling Object Members	Macro Object Members
Since	2018.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var description = macro.description; // get the description of the macro
4 ...
5 // Add additional code

```

Macro.attributes



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The defined attributes of the macro. For information about record macros, see the help topic Overview of Record Action and Macro APIs .
Type	Object
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Parent Object	record.Macro
Sibling Object Members	Macro Object Members
Since	2018.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var attributes = macro.attributes; // get the attributes of the macro
4 ...
5 // Add additional code

```

record.Record



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the record module when you want to work with NetSuite records.

Object Description	Encapsulates a NetSuite record. There are two modes you can operate in when you create, copy, load, or transform a record with SuiteScript 2.0: standard mode and dynamic mode.
---------------------------	--

	<ul style="list-style-type: none"> ■ When a SuiteScript 2.0 script creates, copies, loads, or transforms a record in standard mode, the record's body fields and sublist line items are not sourced, calculated, and validated until the record is saved (submitted) with Record.save(options). <p>When you work with a record in standard mode, you do not need to set values in any particular order. After submitting the record, NetSuite processes the record's body fields and sublist line items in the correct order, regardless of the organization of your script.</p> <ul style="list-style-type: none"> ■ When a SuiteScript 2.0 script creates, copies, loads, or transforms a record in dynamic mode, the record's body fields and sublist line items are sourced, calculated, and validated in real-time. A record in dynamic mode emulates the behavior of a record in the UI. <p>When you work with a record in dynamic mode, it is important that you set values in the same order you would within the UI. If you fail to do this, your results may not be accurate.</p> <p>The record.create(options), record.copy(options), record.load(options), and record.transform(options) methods work in standard mode by default. If you want these methods to work in dynamic mode, you must pass in a specific argument. See the help topic for the applicable method for more information.</p> <p>Use record.Type enum for multiple records. For help finding a record's internal ID, see the help topic How do I find a record's internal ID?</p> <p>For more information about standard and dynamic modes, see the help topic SuiteScript 2.0 – Standard and Dynamic Modes</p> <p>For a complete list of this object's methods and properties, see Record Object Members.</p>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: '6',
6   isDynamic: true
7 });
8 ...
9 // Add additional code.

```

Record.cancelLine(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Cancels the currently selected line on a sublist. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	The record.Record object that called the method.
Supported Script Types	Client and server-side scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 objRecord.cancelLine({
4   sublistId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.commitLine(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Commits the currently selected line on a sublist. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) When working in standard mode, set a sublist field using Record.setSublistValue(options) .
Returns	The record.Record object that called the method.
Supported Script Types	Client and server-side scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.ignoreRecalc	boolean true false	optional	If set to true, scripting recalculation is ignored.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.commitLine({
4     sublistId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.executeMacro(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs macro operation and returns its result in a plain JavaScript object. For information about record macros, see the help topic Overview of Record Action and Macro APIs .
---------------------------	---

Returns	An object with the macro results or null.
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2018.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The macro ID.
options.params	Object	optional	The macro arguments.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 if ('calculateTax' in macros) {
4     macros.calculateTax();
5 }
6 ...
7 // Add additional code

```

Record.executeMacro.promise(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs macro operation and returns its result in a plain JavaScript object. For information about record macros, see the help topic Overview of Record Action and Macro APIs . Note: The parameters and errors thrown for this method are the same as those for <code>Record.executeMacro(options)</code> . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client-side scripts For additional information, see the help topic SuiteScript 2.0 Client Script Type .

Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The macro ID.
options.params	Object	optional	The macro arguments.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 if ('calculateTax' in macros) {
4     macros.calculateTax.promise();
5 }
6 ...
7 // Add additional code

```

Record.findMatrixSublistLineWithValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the line number of the first instance where a specified value is found in a specified column of the matrix. Note that line and column indexing begins at 0 with SuiteScript 2.0. (dynamic and standard modes — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	number
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

i Note:	The options parameter is a JavaScript object.
----------------	---

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2016.2
options.fieldId	string	required	The ID of the matrix field. See, How do I find a field's internal ID?	2016.2
options.value	number	required	The value to search for.	2016.2
options.column	number	required	The column number of the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
--

```

1 // Add additional code.
2 ...
3 var lineNumber = objRecord.findMatrixSublistLineWithValue({
4   sublistId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.findSublistLineWithValue(options)

i Note:	The content in this help topic pertains to SuiteScript 2.0.
----------------	---

Method Description	Returns the line number for the first occurrence of a field value in a sublist. Note that line indexing begins at 0 with SuiteScript 2.0. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	A line number as a number, or -1 if not found.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.value	number Date string array boolean true false	optional	The value to search for.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or not defined.

Syntax

The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var lineNumber = objRecord.findSublistLineWithValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   value: 233
7 });
8 ...
9 // Add additional code.

```

Record.getCurrentMatrixSublistValue(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the value for the currently selected line in the matrix.
---------------------------	---

	(dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Gets a numeric value for rate and ratehighprecision fields.
Returns	number Date string array boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of the matrix field. See, How do I find a field's internal ID?	2015.2
options.column	number	required	The column number for the matrix field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
---	--

```

1 // Add additional code.
2 ...
3 var matrixValue = objRecord.getCurrentMatrixSublistValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   column: 12
7 });
8 ...

```

```
9 | // Add additional code.
```

Record.getCurrentSublistField(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns metadata about a sublist field. (dynamic mode only— see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Field
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```
1 | // Add additional code.
2 | ...
```

```

3 | var sublistFieldMetadata = objRecord.getCurrentSublistField({
4 |   sublistId: 'item',
5 |   fieldId: 'item',
6 | });
7 | ...
8 | // Add additional code.

```

Record.getCurrentSublistIndex(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the line number of the currently selected line. Note that line indexing begins at 0 with SuiteScript 2.0. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	number
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 | // Add additional code.

```

```

2 ...
3 var currIndex = objRecord.getCurrentSublistIndex({
4   sublistId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.getCurrentSublistSubrecord(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the subrecord for the associated sublist field on the current line. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See the help topic How do I find a field's internal ID? The fieldId is inside the sublist. It works if the current sublist contains subrecord fields. You can use Record.getCurrentSublistField(options) to get all metadata about a sublist field.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```
1 // Add additional code.
```

```

2 ...
3 var objSubrecord = objRecord.getCurrentSublistSubrecord({
4   sublistId: 'item',
5   fieldId: 'item'
6 });
7 ...
8 // Add additional code.

```

Record.getCurrentSublistText(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a text representation of the field value in the currently selected line. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Gets a string value with a "%" for rate and ratehighprecision fields.
Returns	string Note: For multiselect fields, returns an array.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Error Code	Thrown If
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var fieldName = objRecord.getCurrentSublistText({
4     sublistId: 'item',
5     fieldId: 'item'
6 });
7 ...
8 // Add additional code.

```

Record.getCurrentSublistValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a sublist field on the currently selected sublist line. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	number Date string array boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field.	2015.2

Parameter	Type	Required / Optional	Description	Since
			See, How do I find a field's internal ID?	

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistValue = objRecord.getCurrentSublistValue({
4   sublistId: 'item',
5   fieldId: 'item'
6 });
7 ...
8 // Add additional code.

```

Record.getField(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a field object from a record. (dynamic and standard modes — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Field
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field.	2015.2

Parameter	Type	Required / Optional	Description	Since
			See, How do I find a field's internal ID?	

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objField = objRecord.getField({
4   fieldId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.getFields()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the body field names (internal ids) of all the fields in the record, including machine header field and matrix header fields. (dynamic and standard modes — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	string[]
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objFields = objRecord.getFields();
4 ...
5 // Add additional code.

```

Record.getLineCount(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the number of lines in a sublist. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	number
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var numLines = objRecord.getLineCount({
4     sublistId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.getMacro(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Provides a macro to be executed.
---------------------------	----------------------------------

	For information about record macros, see the help topic Overview of Record Action and Macro APIs .
Returns	Function to be executed for the macro.
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The macro ID.

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.
SSS_INVALID_MACRO_ID	A macro does not exist on the record.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var macro = recordObj.getMacro({id: 'calculateTax'});
4 ...
5 // Add additional code

```

Record.getMacros(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Provides a plain JavaScript object of available macro objects defined for a record type, indexed by the Macro ID. The object returns one or more <code>record.Macro</code> objects. If there are no macros available for the specified record type, an empty object is returned.
---------------------------	--

	For information about record macros, see the help topic Overview of Record Action and Macro APIs .
Returns	Object
Supported Script Types	Client and server scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2018.2

Errors

Error Code	Thrown If
SSS_INVALID_RECORD_TYPE	The specified record type is invalid.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var macroList = recordObj.getMacros();
4 ...
5 // Add additional code

```

Record.getMatrixHeaderCount(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the number of columns for the specified matrix. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	number
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note:	The options parameter is a JavaScript object.
--------------	---

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	<p>The internal ID of the sublist that contains the matrix.</p> <p>This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser.</p>	2015.2
options.fieldId	string	required	<p>The internal ID of the matrix field.</p> <p>See, How do I find a field's internal ID?</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
-------------------	--

```

1 // Add additional code.
2 ....
3 var numLines = objRecord.getMatrixHeaderCount({
4     sublistId: 'item',
5     fieldId: 'item'
6 });
7 ...
8 // Add additional code.

```

Record.getMatrixHeaderField(options)

Note:	The content in this help topic pertains to SuiteScript 2.0.
Method Description	Gets the field for the specified header in the matrix. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Field
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

i	Note: The options parameter is a JavaScript object.
----------	--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of the matrix field. See, How do I find a field's internal ID?	2015.2
options.column	number	required	The column number for the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
--

```

1 // Add additional code.
2 ...
3 var objField = objRecord.getMatrixHeaderField({
4   sublistId: 'item',
5   fieldId: 'item',
6   column: 12
7 });
8 ...
9 // Add additional code.

```

Record.getMatrixHeaderValue(options)

i	Note: The content in this help topic pertains to SuiteScript 2.0.
----------	--

Method Description	Gets the value for the associated header in the matrix. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Gets a numeric value for rate and ratehighprecision fields.
Returns	number Date string array boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of the matrix field. See, How do I find a field's internal ID?	2015.2
options.column	number	required	The column number for the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var value = objRecord.getMatrixHeaderValue({
4     sublistId: 'item',
5     fieldId: 'item',
6     column: 12
7 });
8 ...
9 // Add additional code.

```

Record.getMatrixSublistField(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the field for the specified sublist in the matrix.
---------------------------	---

	(standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	<code>record.Field</code>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
<code>options.fieldId</code>	string	required	The internal ID of the matrix field. See, How do I find a field's internal ID?	2015.2
<code>options.column</code>	number	required	The column number for the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2
<code>options.line</code>	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
<code>SSS_MISSING_REQD_ARGUMENT</code>	A required argument is missing or undefined.

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
---	--

```

1 // Add additional code.
2 ...
3 var objField = objRecord.getMatrixSublistField({
4   sublistId: 'item',
5   fieldId: 'item',
6   column: 12,

```

```

7   line: 3
8 });
9 ...
10 // Add additional code.

```

Record.getMatrixSublistValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the value for the associated field in the matrix. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	number Date string array boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of the matrix field. See, How do I find a field's internal ID?	2015.2
options.column	number	required	The column number for the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var value = objRecord.getMatrixSublistValue({
4     sublistId: 'item',
5     fieldId: 'item',
6     column: 12,
7     line: 3
8 });
9 ...
10 // Add additional code.

```

Record.getSublist(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the specified sublist. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Sublist
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```
1 // Add additional code.
```

```

2 ...
3 var objSublist = objRecord.getSublist({
4   sublistId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.getSublists()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns all the names of all the sublists. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	string[]
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistName = objRecord.getSublists();
4 ...
5 // Add additional code.

```

Record.getSublistField(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a field object from a sublist. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Field
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module

Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
--

```

1 // Add additional code.
2 ...
3 var objField = objRecord.getSublistField({
4     sublistId: 'item',
5     fieldId: 'item',
6     line: 3
7 });
8 ...
9 // Add additional code.

```

Record.getSublistFields(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Returns all the field names in a sublist. (standard and dynamic mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	string[]

Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

i	Note: The options parameter is a JavaScript object.
----------	--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

! Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples . For other samples, see Sublist.getColumn(options) .

```

1 // Add additional code.
2 ...
3 var field = objRecord.getSublistFields({
4   sublistId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.getSublistSubrecord(options)

i	Note: The content in this help topic pertains to SuiteScript 2.0.
----------	--

Method Description	Gets the subrecord associated with a sublist field. (standard mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) When working in dynamic mode, get a sublist subrecord using the following methods: <ol style="list-style-type: none"> 1. Record.selectLine(options) 2. Record.hasCurrentSublistSubrecord(options)
---------------------------	--

	3. Record.getCurrentSublistSubrecord(options)
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.line	number	required	The line number for the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objSubRecord = objRecord.getSublistSubrecord({
4   sublistId: 'item',
5   fieldId: 'item',
6   line: 3
7 });
8 ...
9 // Add additional code.

```

Record.getSublistText(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a sublist field in a text representation.
---------------------------	--

	(standard mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Gets a string value with a "%" for rate and ratehighprecision fields.
Returns	string Note: For multiselect fields, returns an array.
Supported Script Types	Client and server-side scripts. For more information, see the help topic SuiteScript 2.0 Script Types . Limitations exist on how this method can be used in standard (deferredDynamic) mode. For details, refer to the description of the SSS_INVALID_API_USAGE error code in the Errors table. In dynamic mode, you can use getSublistText() without limitation.
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
SSS_INVALID_API_USAGE	Invoked in certain cases when deferredDynamic mode is being used. For example, if Record.isDynamic is set to false, this error can be invoked in both of the following situations: <ul style="list-style-type: none"> ▪ If the record object was created by <code>record.copy()</code>, <code>record.create()</code>, or <code>record.transform()</code>, and the script attempts to use <code>getSublistText()</code> without first using <code>setSublistText()</code> for the same field. ▪ If the record object was created by <code>record.load()</code>, and the script uses <code>setSublistValue()</code> on a field before using <code>getSublistText()</code> for the same field.

Error Code	Thrown If
	<p>This guidance also affects user event scripts that instantiate records by using the newRecord or oldRecord object provided by the script context. These records always use deferredDynamic mode. For that reason, this error appears in both of the following situations:</p> <ul style="list-style-type: none"> ■ When a user event script executes on a record that is being newly created, and the script attempts to use getSublistText() without first using setSublistText() for the same field. ■ When a user event script executes on an existing record, and the script uses setSublistValue() on a field before using getSublistText() for the same field. <p>For more information, see the help topic SuiteScript 2.0 – Standard and Dynamic Modes.</p>
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistFieldName = objRecord.getSublistText({
4     sublistId: 'item',
5     fieldId: 'item',
6     line: 3
7 });
8 ...
9 // Add additional code.

```

Record.getSublistValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a sublist field. (dynamic and standard modes — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Gets a numeric value for rate and ratehighprecision fields.
Returns	number Date string array boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
SSS_INVALID_API_USAGE	Invoked prior to using setSublistValue in standard record mode.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var sublistFieldValue = objRecord.getSublistValue({
4     sublistId: 'item',
5     fieldId: 'item',
6     line: 3
7 });
8 ...
9 // Add additional code.

```

Record.getSubrecord(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the subrecord for the associated field. This method is not available for subrecords. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
---------------------------	--

Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See, How do I find a field's internal ID?	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
FIELD_1_IS_NOT_A_SUBRECORD_FIELD	The specified field is not a subrecord field.
FIELD_1_IS_DISABLED_YOU_CANNOT_APPLY_SUBRECORD_OPERATION_ON_THIS_FIELD	The specified field is disabled.

Syntax

 Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
--

```

1 // Add additional code.
2 ...
3 var sublistFieldValue = objRecord.getSubrecord({
4   fieldId: 'idnumber'
5 });
6 ...
7 // Add additional code.

```

Record.getText(options)

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Method Description	Returns the text representation of a field value.
---------------------------	---

	(dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Gets a string value with a "%" for rate and ratehighprecision fields.
Returns	string  Note: For multiselect fields, returns an array.
Supported Script Types	Client and server-side scripts. For more information, see the help topic SuiteScript 2.0 Script Types . In dynamic mode, you can use getText() without limitation but, in standard mode, limitations exist. In standard mode, you can use this method only in the following cases: <ul style="list-style-type: none">■ You can use getText() on any field where the script has already used setText().■ If you are loading or copying a record, you can use getText on any field except those where the script has already changed the value by using setValue(). For more details, refer to the description of the SSS_INVALID_API_USAGE error code in the Errors table.
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See, How do I find a field's internal ID?	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_API_USAGE	Invoked in certain cases when standard mode is being used. For example, if <code>Record.isDynamic</code> is set to false, the SSS_INVALID_API_USAGE error can be invoked in the following situations: <ul style="list-style-type: none">■ If the record object was created by <code>record.create()</code> or <code>record.transform()</code>, and the script attempts to use <code>getText()</code> without first using <code>setText()</code> for the same field.■ The record object was created by <code>record.copy()</code> or <code>record.load()</code>, and the script uses <code>setValue()</code> on a field before using <code>getText()</code> for the same field. Similar guidance affects user event scripts that instantiate records by using the <code>newRecord</code> or <code>oldRecord</code> object provided by the script context. In these cases,

Error Code	Thrown If
	<p>standard mode is always used. For that reason, the SSS_INVALID_API_USAGE error appears when a user event executes on one of these objects in the following situations:</p> <ul style="list-style-type: none"> ▪ When the script executes on a record that is being created, and the script attempts to use getText() without first using setText() for the same field. ▪ When the script executes on an existing record or on a record being created through copying, and the script uses setValue() on a field before using getText() for the same field.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Syntax Sample 1
2 // Add additional code.
3 ...
4 var fieldidname = objRecord.getText({
5   fieldId: 'item'
6 });
7 ...
8 // Add additional code.
9
10 // Syntax Sample 2
11 // Add additional code.
12 ...
13 myString = 'Date is: ' + record.getText({fieldId: 'datechanged'});
14 // "Date is: 3/27/2017 9:55:38am"
15 ...
16 // Add additional code

```

Record.getValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a field. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Gets a numeric value for rate and ratehighprecision fields.
Returns	number Date string array boolean true false Returns a JavaScript Date object for date/time field queries. To return a string for date/time field queries, use Record.getText(options) . Date/time fields: DATE, DATETIME, DATETIMETZ, TIMEOFTDAY.
	 Note: If the returned date object is implicitly converted to a string, the value is converted using the browser's setting for time zone.
	All fields of type CHECKBOX return true or false.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

i	Note: The options parameter is a JavaScript object.
----------	--

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See, How do I find a field's internal ID?	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_API_USAGE	Invoked in standard mode, if you use setText on a field and then use getValue on the same field.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
--

```

1 // Add additional code.
2 ...
3 var value = objRecord.getValue({
4   fieldId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.hasCurrentSublistSubrecord(options)

i	Note: The content in this help topic pertains to SuiteScript 2.0.
----------	--

Method Description	Returns a value indicating whether the associated sublist field has a subrecord on the current line. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a subrecord. See, How do I find a field's internal ID?	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var hasSubrecord = objRecord.hasCurrentSublistSubrecord({
4     sublistId: 'item',
5     fieldId: 'item'
6 });
7 ...
8 // Add additional code.

```

Record.hasSublistSubrecord(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a value indicating whether the associated sublist field contains a subrecord. This method is not available for subrecords. (standard mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a subrecord. See, How do I find a field's internal ID?	2015.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var hasSubrecord = objRecord.hasSublistSubrecord({
4     sublistId: 'item',
5     fieldId: 'item',
6     line: 3
7 });
8 ...
9 // Add additional code.

```

Record.hasSubrecord(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a value indicating whether the field contains a subrecord. This method is not available for subrecords. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of the field that may contain a subrecord. See, How do I find a field's internal ID?	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var hasSubrecord = objRecord.hasSubrecord({
4     fieldId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.insertLine(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Inserts a sublist line. When you insert a line with this method, all succeeding lines are moved and the total line count is increased. Essentially, succeeding lines are committed to a new sublist line with a new line number. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members

Since	2015.2
-------	--------

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.line	number	required	The line number to insert. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2
options.ignoreRecalc	boolean true false	optional	If set to true, scripting recalculation is ignored. The default value is false.	2015.2
options.beforeLineInstanceId	string	required	The line instance ID. Use this parameter to specify where to insert the line.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example that uses `insertLine()`, see the help topic [Example: Creating a Landed Cost Sublist Subrecord](#).

```

1 // Add additional code.
2 ...
3 objRecord.insertLine({
4     sublistId: 'attendee',
5     line: 2,
6 });
7 objRecord.setCurrentSublistValue({
8     sublistId: 'attendee',
9     fieldId: 'attendee',
10    value: 838
11 });
12 objRecord.commitLine({
13     sublistId: 'attendee'
14 });
15 ...
16 // Add additional code.

```

For script examples that use other N/record methods, see record Module Script Samples.

Record.removeCurrentSublistSubrecord(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the subrecord for the associated sublist field on the current line. This method is not available for subrecords. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	<code>record.Record</code>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
<code>options.fieldId</code>	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.removeCurrentSublistSubrecord({
4   sublistId: 'item',
5   fieldId: 'item'
6 });
7 ...
8 // Add additional code.

```

Record.removeLine(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes a sublist line. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	<code>record.Record</code>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
<code>options.line</code>	number	required	The line number of the sublist to remove. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2
<code>options.ignoreRecalc</code>	boolean true false	optional	If set to true, scripting recalculation is ignored. The default value is false.	2015.2
<code>options.lineInstanceId</code>	string	required	The line instance ID. Use this parameter to specify where to remove the line.	2015.2

Errors

Error Code	Thrown If
<code>SSS_MISSING_REQD_ARGUMENT</code>	A required argument is missing or undefined.
<code>SSS_INVALID_SUBLIST_OPERATION</code>	A required argument is invalid or the sublist is not editable.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 // The following code snippet assumes that there are at least three committed lines present in the item sublist.
3 ...
4 objRecord.removeLine({
5   sublistId: 'item',
6   line: 3,
7   ignoreRecalc: true
8 });
9 ...
10 // Add additional code.

```

Record.removeSublistSubrecord(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the sublist record for the associated sublist field. (standard mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) When working in dynamic mode, remove a sublist subrecord using the following methods: <ol style="list-style-type: none"> 1. Record.selectLine(options) 2. Record.hasCurrentSublistSubrecord(options) 3. Record.removeCurrentSublistSubrecord(options) 4. Record.commitLine(options)
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field.	2015.2

Parameter	Type	Required / Optional	Description	Since
			See, How do I find a field's internal ID?	
options.line	number	required	The line number in the sublist that contains the subrecord to remove. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.removeSublistSubrecord({
4   sublistId: 'item',
5   fieldid: 'item',
6   line: 3
7 });
8 ...
9 // Add additional code.

```

Record.removeSubrecord(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the subrecord for the associated field. This method is not available for subrecords. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See, How do I find a field's internal ID?	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.removeSubrecord({
4   fieldid: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.save(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Submits a new record or saves edits to an existing record. When working with records in standard mode, you must submit and then load the record to obtain sourced, validated, and calculated field values. This method is not available to subrecords.
	Note: This method has an asynchronous counterpart you can use with client scripts. See Record.save.promise(options) .
Returns	A number representing the internal ID of the new or updated record.
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Transaction records: 20 units Custom records: 4 units All other records: 10 units
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.enableSourcing	boolean true false	optional	Enables sourcing during the record update. If set to true, sources dependent field information for empty fields. Defaults to false – dependent field values are not sourced.	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p>⚠️ Important: This parameter applies to records in standard mode only. When working with records in dynamic mode, field values are always sourced and the value you provide for enableSourcing is ignored. See the help topic SuiteScript 2.0 – Standard and Dynamic Modes.</p>	
options.ignoreMandatoryFields	boolean true false	optional	<p>Disables mandatory field validation for this save operation.</p> <p>If set to true, all standard and custom fields that were made mandatory through customization are ignored. All fields that were made mandatory through company preferences are also ignored.</p> <p>By default, this parameter is false.</p> <p>⚠️ Important: Use the ignoreMandatoryFields argument with caution. This argument should be used mostly with Scheduled scripts, rather than User Event scripts. This ensures that UI users do not bypass the business logic enforced through form customization.</p>	2015.2

Syntax

⚠️ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var recordId = objRecord.save({
4   enableSourcing: true,
5   ignoreMandatoryFields: true
6 });
7 ...
8 // Add additional code.

```

Record.save.promise(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Submits a new record asynchronously or saves edits to an existing record asynchronously. This method is not available to subrecords.
	<p>i Note: The parameters and errors thrown for this method are the same as those for Record.save(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Supported Script Types	Client-side scripts

	For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	Transaction records: 20 units Custom records: 4 units All other records: 10 units
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.enableSourcing	boolean true false	optional	<p>Enables sourcing during the record update.</p> <p>If set to true, sources dependent field information for empty fields.</p> <p>Defaults to false – dependent field values are not sourced.</p> <p>Important: This parameter applies to records in standard mode only. When working with records in dynamic mode, field values are always sourced and the value you provide for enableSourcing is ignored. See the help topic SuiteScript 2.0 – Standard and Dynamic Modes.</p>	2015.2
options.ignoreMandatoryFields	boolean true false	optional	<p>Disables mandatory field validation for this save operation.</p> <p>If set to true, all standard and custom fields that were made mandatory through customization are ignored. All fields that were made mandatory through company preferences are also ignored.</p> <p>By default, this parameter is false.</p> <p>Important: Use the ignoreMandatoryFields argument with caution. This argument should be used mostly with Scheduled scripts, rather than User Event scripts. This ensures that UI users do not bypass the business logic enforced through form customization.</p>	2015.2

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```
1 | // Add additional code.
```

```

2 ...
3 var recordId = objRecord.save.promise({
4   enableSourcing: true,
5   ignoreMandatoryFields: true
6 });
7 ...
8 // Add additional code.

```

Record.selectLine(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Selects an existing line in a sublist. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) When working in standard mode, set a sublist field using Record.setSublistValue(options) .
Returns	<code>record.Record</code>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
<code>options.line</code>	number	required	The line number to select in the sublist. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2

Errors

Error Code	Thrown If
<code>SSS_MISSING_REQD_ARGUMENT</code>	A required argument is missing or undefined.
<code>SSS_INVALID_SUBLIST_OPERATION</code>	A required argument is invalid or the sublist is not editable.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var lineNum = objRecord.selectLine({
4     sublistId: 'item',
5     line: 3
6 });
7 ...
8 // Add additional code.

```

Record.selectNewLine(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Selects a new line at the end of a sublist. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var lineNum = objRecord.selectNewLine({
4     sublistId: 'item'
5 });
6 ...
7 // Add additional code.

```

Record.setCurrentMatrixSublistValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the line currently selected in the matrix. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Sets a string value with a "%" for rate and ratehighprecision fields.
Returns	<code>record.Record</code>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.sublistId</code>	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
<code>options.fieldId</code>	string	required	The internal ID of the matrix field. See, How do I find a field's internal ID?	2015.2
<code>options.column</code>	number	required	The column number for the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2

Parameter	Type	Required / Optional	Description	Since
options.value	number Date string array boolean true false	required	<p>The value to set the field to.</p> <p>The value type must correspond to the field type being set. For example:</p> <ul style="list-style-type: none"> ■ Text, Radio and Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2015.2
options.ignoreFieldChange	boolean true false	optional	<p>If set to true, the field change and the secondary event is ignored.</p> <p>By default, this value is false.</p>	2015.2

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.setCurrentMatrixSublistValue({
4     sublistId: 'item',
5     fieldId: 'item',
6     column: 3,
7     value: false,
8     ignoreFieldChange: true,
9     forceSyncSourcing: true
10 });
11 ...
12 // Add additional code.

```

Record.setCurrentSublistText(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Sets the value for the field in the currently selected line by a text representation.</p> <p>(dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)</p> <p>Sets a string value with a "%" for rate and ratehighprecision fields.</p>
---------------------------	--

Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.text	string	required	The text to set the value to.	2015.2
options.ignoreFieldChange	boolean true false	optional	If set to true, the field change and the secondary event is ignored. By default, this value is false.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
A_SCRIPT_IS_ATTEMPTING_TO_EDIT_THE_1_SUBLIST_THIS_SUBLIST_IS_CURRENTLY_IN_READONLY_MODE_AND_CANNOT_BE_EDITED_CALL_YOUR_NETSUITE_ADMINISTRATOR_TO_DISABLE_THIS_SCRIPT_IF_YOU_NEED_TO_SUBMIT_THIS_RECORD	A user tries to edit a read-only sublist field.

Syntax

 Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
--

```

1 // Add additional code.
2 ...
3 objRecord.setCurrentSublistText({
4   sublistId: 'item',

```

```

5   fieldId: 'item',
6   text: 'value',
7   ignoreFieldChange: true
8 });
9 ...
10 // Add additional code.

```

Record.setCurrentSublistValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the field in the currently selected line. (dynamic mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) When working in standard mode, set a sublist field using Record.setSublistValue(options) .
	 Important: When you edit a sublist line with SuiteScript, it triggers an internal validation of the sublist line. If the line validation fails, the script also fails. For example, if your script edits a closed catch up period, the validation fails and prevents SuiteScript from editing the closed catch up period.
	Sets a numeric value for rate and ratehighprecision fields.
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.value	number Date string array	required	The value to set the field to. The value type must correspond to the field type being set. For example:	2015.2

Parameter	Type	Required / Optional	Description	Since
	boolean true false		<ul style="list-style-type: none"> ■ Text, Radio and Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	
options.ignoreFieldChange	boolean true false	optional	If set to true, the field change and the secondary event is ignored. By default, this value is false.	2015.2

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
A_SCRIPT_IS_ATTEMPTING_TO_EDIT_THE_1 sublist THIS sublist IS CURRENTLY IN READONLY_MODE_AND_CANNOT_BE_EDITED_CALL_YOUR_NETSUITE_ADMINISTRATOR_TO_DISABLE_THIS_SCRIPT_IF_YOU_NEED_TO_SUBMIT_THIS_RECORD	A user tries to edit a read-only sublist field.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.setCurrentSublistValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   value: true,
7   ignoreFieldChange: true
8 });
9 ...
10 // Add additional code.

```

Record.setMatrixHeaderValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the associated header in the matrix. (dynamic and standard modes — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Sets a numeric value for rate and ratehighprecision fields.
Returns	record.Record

Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of the matrix field. See, How do I find a field's internal ID?	2015.2
options.column	number	required	The column number for the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2
options.value	number Date string array boolean true false	required	The value to set the field to. The value type must correspond to the field type being set. For example: <ul style="list-style-type: none">■ Text, Radio and Select fields accept string values.■ Checkbox fields accept Boolean values.■ Date and DateTime fields accept Date values.■ Integer, Float, Currency and Percent fields accept number values.	2015.2
options.ignoreFieldChange	boolean true false	optional	If set to true, the field change and the secondary event is ignored. By default, this value is false.	2015.2

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.setMatrixHeaderValue({
4     sublistId: 'item',
5     fieldId: 'item',
6     column: 3,
7     value: false,
8     ignoreFieldChange: true,
9     forceSyncSourcing: true
10 });
11 ...
12 // Add additional code.

```

Record.setMatrixSublistValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value for the associated field in the matrix. (standard mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Sets a numeric value for rate and ratehighprecision fields.
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist that contains the matrix. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of the matrix field. See, How do I find a field's internal ID?	2015.2

Parameter	Type	Required / Optional	Description	Since
options.column	number	required	The column number for the field. Note that column indexing begins at 0 with SuiteScript 2.0.	2015.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2
options.value	number Date string array boolean true false	required	<p>The value to set the field to.</p> <p>The value type must correspond to the field type being set. For example:</p> <ul style="list-style-type: none"> ■ Text, Radio and Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2015.2

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.setMatrixSublistValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   column: 12,
7   line: 3,
8   value: true
9 });
10 ...
11 // Add additional code.

```

Record.setSublistText(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of a sublist field by a text representation. (standard mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) When working in dynamic mode, set a sublist field text using the following methods:
---------------------------	--

	<ol style="list-style-type: none"> 1. Record.selectLine(options) 2. Record.setCurrentSublistText(options) 3. Record.commitLine(options) <p>Sets a string value with a "%" for rate and ratehighprecision fields.</p>
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2
options.line	number	required	The line number for the field. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2
options.text	string	required	The text to set the value to.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
---	--

```

1 // Add additional code.
2 ...
3 objRecord.setCurrentSublistText({
4   sublistId: 'item',

```

```

5   fieldId: 'item',
6   line: 3,
7   text: 'value'
8 });
9 ...
10 // Add additional code.

```

Record.setSublistValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of a sublist field. (standard mode only — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) When working in dynamic mode, set a sublist field value using the following methods: <ol style="list-style-type: none">1. Record.selectLine(options)2. Record.setCurrentSublistValue(options)3. Record.commitLine(options)
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.sublistId	string	required	The internal ID of the sublist. This value is displayed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .	2015.2
options.fieldId	string	required	The internal ID of a standard or custom sublist field. See, How do I find a field's internal ID?	2015.2

Parameter	Type	Required / Optional	Description	Since
options.line	number	required	The line number of the sublist. Note that line indexing begins at 0 with SuiteScript 2.0.	2015.2
options.value	number Date string array boolean true false	required	<p>The value to set the sublist field to.</p> <p>The value type must correspond to the field type being set. For example:</p> <ul style="list-style-type: none"> ■ Text, Radio and Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2015.2

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.
SSS_INVALID_SUBLIST_OPERATION	A required argument is invalid or the sublist is not editable.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.setSublistValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   line: 3,
7   value: true
8 });
9 ...
10 // Add additional code.

```

Record.setText(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of the field by a text representation. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes) Sets a string value with a "%" for rate and ratehighprecision fields.
Returns	record.Record

Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of a standard or custom body field. See, How do I find a field's internal ID?	2015.2
options.text	string array	required	The text or texts to change the field value to. <ul style="list-style-type: none"> ■ If the field type is multiselect: <ul style="list-style-type: none"> □ This parameter accepts an array of string values. □ This parameter accepts a null value. Passing in null deselects all currently selected values. ■ If the field type is not multiselect, this parameter accepts only a single string value. 	2015.2
options.ignoreFieldChange	boolean true false	optional	If set to true, the field change and the secondary event is ignored. By default, this value is false.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
--

```

1 // Add additional code.
2 ...
3 objRecord.setText({
4   fieldId: 'item',
5   text: 'value',
6   ignoreFieldChange: true

```

```

7 | });
8 | ...
9 | // Add additional code.

```

Record.setValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of a field. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Returns	<code>record.Record</code>
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.fieldId</code>	<code>string</code>	required	The internal ID of a standard or custom body field. See, How do I find a field's internal ID?	2015.2
<code>options.value</code>	<code>number Date string array boolean true false</code>	required	<p>The value to set the field to.</p> <p>The value type must correspond to the field type being set. For example:</p> <ul style="list-style-type: none"> ■ Text, Radio, Select and Multi-Select fields accept string and array of values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2015.2
<code>options.ignoreFieldChange</code>	<code>boolean true false</code>	optional	If set to true, the field change and the secondary event is ignored.	2015.2

Parameter	Type	Required / Optional	Description	Since
			By default, this value is false.	

Errors

Error Code	Thrown If
INVALID_FLD_VALUE	The options.value type does not match the field type.
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 objRecord.setValue({
4   fieldId: 'item',
5   value: true,
6   ignoreFieldChange: true
7 });
8 ...
9 // Add additional code.

```

Record.id

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of a specific record. This property is not available to subrecords. (dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)
Type	number (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Record.isDynamic

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the record is in dynamic or standard mode.
-----------------------------	--

	<ul style="list-style-type: none"> ■ If set to true, the record is currently in dynamic mode. If set to false, the record is currently in standard mode. <ul style="list-style-type: none"> □ When a SuiteScript 2.0 script creates, copies, loads, or transforms a record in standard mode, the record's body fields and sublist line items are not sourced, calculated, and validated until the record is saved (submitted) with Record.save(options). When you work with a record in standard mode, you do not need to set values in any particular order. After submitting the record, NetSuite processes the record's body fields and sublist line items in the correct order, regardless of the organization of your script. □ When a SuiteScript 2.0 script creates, copies, loads, or transforms a record in dynamic mode, the record's body fields and sublist line items are sourced, calculated, and validated in real-time. A record in dynamic mode emulates the behavior of a record in the UI. When you work with a record in dynamic mode, it is important that you set values in the same order you would within the UI. If you fail to do this, your results may not be accurate. <p>This value is set when the record is created or accessed.</p> <p>(dynamic and standard mode — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)</p>
Type	boolean true false (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 if (record.isDynamic) {
4   ...
5 }
6 ...
7 // Add additional code.

```

Record.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The record type. Note the following: <ul style="list-style-type: none"> ■ When working with an instance of a standard NetSuite record type, set this value by using the record.Type enum.
-----------------------------	--

	<ul style="list-style-type: none"> When working with an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. <p>This property is not available to subrecords.</p> <p>(dynamic and standard modes — see the help topic SuiteScript 2.0 – Standard and Dynamic Modes)</p>
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Record Object Members
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3
4 // Start the process of creating an employee record.
5
6 var employeeRecord = record.create({
7     type: record.Type.EMPLOYEE,
8     isDynamic: true
9 });
10
11 // Start the process of creating an instance of a custom record type.
12
13 var customRecord = record.create({
14     type: 'customrecord_book',
15     isDynamic: true
16 });
17
18 ...
19 // Add additional code.

```



Note: Supported standard record types are described in the [SuiteScript Records Browser](#). Refer also to [SuiteScript Supported Records](#). For help working with custom record types, see the help topic [Custom Record](#).

record.Sublist



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a sublist on a standard or custom record. For a complete list of this object's methods and properties, see Sublist Object Members .
---------------------------	---

Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11 if(objSublist.type === 'inlineeditor'){
12   //Perform an action
13 }
14 ...
15 // Add additional code.
```

Sublist.getColumn(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a column in the sublist.
Returns	record.Column
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/record Module
Sibling Object Members	Sublist Object Members
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fieldId	string	required	The internal ID of the column field in the sublist.	2015.2

Parameter	Type	Required / Optional	Description	Since
			See, How do I find a field's internal ID?	

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

Example 1

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4     type: record.Type.SALES_ORDER,
5     id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9     sublistId: 'item'
10 });
11
12 var objColumn = objSublist.getColumn({
13     fieldId: 'item'
14 });
15
16 if(objColumn.type === 'checkbox'){
17     //Perform an action
18 }
19 ...
20 // Add additional code.

```

Example 2

This example loops through each line of the items sublist on a sales order record.

```

1 // Add additional code
2 ...
3 onRequest: function(context) {
4     var recordObj = record.create({type: record.Type.SALES_ORDER});
5     var columnList = recordObj.getSublistFields({sublistId: 'item'});
6     var sublistObj = recordObj.getSublist({sublistId: 'item'});
7
8     for (var i = 0; i < columnList.length; i++) {
9         var columnId = columnList[i];
10        var columnObj = sublistObj.getColumn({fieldId: columnId});
11        if (columnObj !== null) {
12            log.debug('[Column id] = ' + columnObj.id + ' [Column type] = ' + columnObj.type +
13                ' [Column label] = ' + columnObj.label);
14        }
15    }
16 ...
17 ...
18 // Add additional code

```

Sublist.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the internal ID of the sublist.
-----------------------------	---

Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Sublist Object Members
Since	2015.2

Syntax

The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4     type: record.Type.SALES_ORDER,
5     id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9     sublistId: 'item'
10 });
11 //Perform an action with the objSublist.id value
12 ...
13 // Add additional code.
```

Sublist.isChanged



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the sublist has changed on the record form.
Type	boolean true false (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Sublist Object Members
Since	2015.2

Syntax

The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4     type: record.Type.SALES_ORDER,
5     id: 275
6 });
```

```

7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 if(objSublist.isChanged){
13   //Perform an action when objSublist.isChanged is true
14 }
15 ...
16 // Add additional code.

```

Sublist.isDisplay



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the sublist is displayed on the record form.
Type	boolean true false
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Sublist Object Members
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 if(objSublist.isDisplay){
13   //Perform an action when objSublist.isDisplay is true
14 }
15 ...
16 // Add additional code.

```

Sublist.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the sublist type. For more information on sublist types, see serverWidget.SublistType .
-----------------------------	---

	Important: Sublist.type will return a lower case string representing the sublist type. For example, inlineeditor not INLINEEDITOR.
Type	string (read-only)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Sibling Object Members	Sublist Object Members
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.load({
4   type: record.Type.SALES_ORDER,
5   id: 275
6 });
7
8 var objSublist = objRecord.getSublist({
9   sublistId: 'item'
10 });
11
12 if(objSublist.type === 'inlineeditor'){
13   //Perform an action
14 }
15 ...
16 // Add additional code.

```

record.attach(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Attaches a record to another record. Note: For the promise version of this method, see record.attach.promise(options) . Note that promises are only supported in client scripts.
Returns	void
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/record Module

Since	2015.2
--------------	--------

Parameters

i	Note: The options parameter is a JavaScript object.
----------	--

Parameter	Type	Required / Optional	Description	Since
options.record	record.Record	required	The record to attach.	2015.2
options.record.type	string	required	The type of record to attach. Set this value using the record.Type enum. To attach a file from the File Cabinet to a record, set type to file.	2015.2
options.record.id	number string	required	The internal ID of the record to attach.	2015.2
options.to	record.Record	required	The record that the options.record gets attached to.	2015.2
options.to.type	string	required	The record type of the record to attach to. Set the value using the record.Type enum. To attach a file from the File Cabinet to a record, set type to file.	2015.2
options.to.id	number string	required	The internal ID of the record to attach to.	2015.2
options.attributes	Object	optional	The name-value pairs containing attributes for the attachment. By default, this value is null.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

!	Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/record Module Script Samples .
----------	--

```
1 // Add additional code.
```

```

2 ...
3 var id = record.attach({
4   record: {
5     type: 'file',
6     id: '200'
7   },
8   to: {
9     type: 'customer',
10    id: '90'
11  }
12 });
13 ...
14 // Add additional code.

```

```

1 record.attach({
2   record: {
3     type: 'contact',
4     id: 2},
5   to: {
6     type: 'customer',
7     id: 3},
8   attributes: {
9     role: 3}
10 });

```

record.attach.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Attaches a record asynchronously to another record.
	 Note: The parameters and errors thrown for this method are the same as those for record.attach(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/record Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.record	record.Record	required	The record to attach.	2015.2
options.record.type	string	required	The type of record to attach.	2015.2

Parameter	Type	Required / Optional	Description	Since
			Set the value using the record.Type enum. To attach a file from the File Cabinet to a record, set type to file.	
options.record.id	number string	required	The internal ID of the record to attach.	2015.2
options.to	record.Record	required	The record that the options.record gets attached to.	2015.2
options.to.type	string	required	The record type of the record to attach to. Set the value using the record.Type enum. To attach a file from the File Cabinet to a record, set type to file.	2015.2
options.to.id	number string	required	The internal ID of the record to attach to.	2015.2
options.attributes	Object	optional	The name-value pairs containing attributes for the attachment. By default, this value is null.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code.
2 ...
3 function attachRecord() {
4
5     var attachRecordPromise = record.attach.promise({
6         record: {
7             type: record.Type.CONTACT,
8             id: '97'
9         },
10        to: {
11            type: record.Type.OPPORTUNITY,
12            id: '16'
13        }
14    });
15
16    attachRecordPromise.then(function() {
17

```

```

18 // Add any other needed logic that shouldn't execute until
19 // after the contact record is attached to the opportunity.
20
21     log.debug({
22         title: 'Record updated',
23         details: 'Attachment successful'
24     });
25
26 }, function(e) {
27     log.error({
28         title: e.name,
29         details: e.message
30     });
31 });
32 }
33 ...
34 // Add additional code.

```

record.copy(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new record by copying an existing record in NetSuite. Note: For the promise version of this method, see record.copy.promise(options) . Note that promises are only supported in client scripts.
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Transaction records: 10 units Custom records: 2 units All other records: 5 units
Module	N/record Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	The record type. Use the following guidelines: <ul style="list-style-type: none"> ■ When copying an instance of a standard NetSuite record type, set this value by using the record.Type enum. 	2015.2

Parameter	Type	Required / Optional	Description	Since
			<ul style="list-style-type: none"> ■ When copying an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	
options.id	number	required	The internal ID of the existing record instance in NetSuite.	2015.2
options.isDynamic	boolean true false	optional	<p>Determines whether the new record is created in dynamic mode.</p> <ul style="list-style-type: none"> ■ If set to true, the new record is created in dynamic mode. ■ If set to false, the new record is created in standard mode. <p>By default, this value is false.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: For additional information on standard and dynamic mode, see record.Record and SuiteScript 2.0 – Standard and Dynamic Modes. </div>	2015.2
options.defaultValues	Object	optional	<p>Name-value pairs containing default values of fields in the new record.</p> <p>By default, this value is null.</p> <p>For a list of available record default values, see N/record Default Values in the NetSuite Help Center.</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.copy({
4   type: record.Type.SALES_ORDER,
5   id: 157,
6   isDynamic: true,
7   defaultValues: {
8     entity: 107
9   }
10 });
11 ...
12 // Add additional code.

```

record.copy.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new record asynchronously by copying an existing record in NetSuite. Note: The parameters and errors thrown for this method are the same as those for record.copy(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	Transaction records: 10 units Custom records: 2 units All other records: 5 units
Module	N/record Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	The record type. Use the following guidelines: <ul style="list-style-type: none">■ When copying an instance of a standard NetSuite record type, set this value by using the record.Type enum.■ When copying an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record.	2015.2
options.id	number	required	The internal ID of the existing record instance in NetSuite.	2015.2
options.isDynamic	boolean true false	optional	Determines whether the new record is created in dynamic mode. <ul style="list-style-type: none">■ If set to true, the new record is created in dynamic mode.■ If set to false, the new record is created in standard mode. By default, this value is false.	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p> Note: For additional information on standard and dynamic mode, see record.Record and SuiteScript 2.0 – Standard and Dynamic Modes.</p>	
options.defaultValues	Object	optional	<p>Name-value pairs containing default values of fields in the new record.</p> <p>By default, this value is null.</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code.
2 ...
3 function copyRecord() {
4
5     // Copy an instance of a standard record type.
6
7     var copyRecordPromise = record.copy.promise({
8         type: record.Type.PHONE_CALL,
9         id: 165
10    });
11
12     // Note: To copy an instance of a custom record type,
13     // use the record type's string ID instead of the record
14     // module's Type enum. For example:
15     // type:'customrecord_feature'
16
17
18
19     copyRecordPromise.then(function(recordObject) {
20
21         recordObject.setValue({
22             fieldId: 'title',
23             value: 'Sprint 5 bug triage'
24        });
25
26         recordObject.setValue({
27             fieldId: 'message',
28             value: 'Please review the PowerPoint prior to the call.'
29        });
30
31     var recordId = recordObject.save();
32
33     // Add any other needed logic that shouldn't execute until
34     // after the record is copied.
35
36     log.debug({
37         title: 'Record saved',
38         details: 'Id of new record: ' + recordId
39    });
40

```

```

41     }, function(e) {
42         log.error({
43             title: e.name,
44             details: e.message
45         });
46     });
47 }
48 ...
49 // Add additional code.

```

record.create(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new record.
	<p>Note: For the promise version of this method, see record.create.promise(options). Note that promises are only supported in client scripts.</p>
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Transaction records: 10 units Custom records: 2 units All other records: 5 units
Module	N/record Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	<p>The record type.</p> <p>This value determines the Record.type property of the record that is created. This property is read-only on an existing record.</p> <p>Use the following guidelines:</p> <ul style="list-style-type: none"> ■ When creating an instance of a standard NetSuite record type, set this value by using the record.Type enum. ■ When creating an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	2015.2

Parameter	Type	Required / Optional	Description	Since
options.isDynamic	boolean true false	optional	<p>Determines whether the new record is created in dynamic mode.</p> <ul style="list-style-type: none"> ▪ If set to true, the new record is created in dynamic mode. ▪ If set to false, the new record is created in standard mode. <p>By default, this value is false.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: For additional information on standard and dynamic mode, see record.Record and SuiteScript 2.0 – Standard and Dynamic Modes. </div>	2015.2
options.defaultValues	Object	optional	<p>Name-value pairs containing default values of fields in the new record.</p> <p>By default, this value is null.</p> <p>For a list of available record default values, see N/record Default Values in the NetSuite Help Center.</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3
4 // Start the process of creating a sales order record.
5
6 var objRecord = record.create({
7   type: record.Type.SALES_ORDER,
8   isDynamic: true,
9   defaultValues: {
10     entity: 87
11   }
12 });
13
14 // Start the process of creating an instance of a custom record type.
15
16 var customRecord = record.create({
17   type: 'customrecord_feature',
18   isDynamic: true
19 });
20
21 // Get two default values
22

```

```

23 | ...
24 | record.create({
25 |   type: record.Type.SALES_ORDER,
26 |   defaultValues: {
27 |     entity: 107,
28 |     subsidiary: 1
29 |   }
30 | })...
31 | // Add additional code.

```

record.create.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new record asynchronously.
	 Note: The parameters and errors thrown for this method are the same as those for record.create(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	Transaction records: 10 units Custom records: 2 units All other records: 5 units
Module	N/record Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	<p>The record type.</p> <p>This value determines the Record.type property of the record that is created. This property is read-only on an existing record.</p> <p>Use the following guidelines:</p> <ul style="list-style-type: none"> ■ When creating an instance of a standard NetSuite record type, set this value by using the record.Type enum. ■ When creating an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	2015.2

Parameter	Type	Required / Optional	Description	Since
options.isDynamic	boolean true false	optional	<p>Determines whether the new record is created in dynamic mode.</p> <ul style="list-style-type: none"> ▪ If set to true, the new record is created in dynamic mode. ▪ If set to false, the new record is created in standard mode. <p>By default, this value is false.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: For additional information on standard and dynamic mode, see record.Record and SuiteScript 2.0 – Standard and Dynamic Modes. </div>	2015.2
options.defaultValues	Object	optional	<p>Name-value pairs containing default values of fields in the new record.</p> <p>By default, this value is null.</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code.
2
3 ...
4 function createRecord() {
5
6   // Create an instance of a standard record record
7   // type.
8
9   var createRecordPromise = record.create.promise({
10     type: record.Type.PHONE_CALL,
11     isDynamic: true
12   });
13
14   // Note: To create an instance of a custom record type,
15   // use the record type's string ID instead of the record
16   // module's Type enum. For example:
17   // type:'customrecord_feature'
18
19   createRecordPromise.then(function(objRecord) {
20     objRecord.setValue({
21       fieldId: 'title',
22       value: 'sprint planning'
23     });
24
25     var recordId = objRecord.save();
26
27
28

```

```

29     log.debug({
30       title: 'New record saved',
31       details: 'New record ID: ' + recordId
32     });
33
34   }, function(e) {
35     log.error({
36       title: 'Unable to create record',
37       details: e.name
38     });
39   });
40 }
41 ...
42 //Add additional code.

```

record.delete(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Deletes a record.
	<p>Note: For the promise version of this method, see record.delete.promise(options). Note that promises are only supported in client scripts.</p>
Returns	The internal ID of the deleted record.Record .
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Transaction records: 20 units Custom records: 4 units All other records: 10 units
Module	N/record Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	<p>The record type. Use the following guidelines:</p> <ul style="list-style-type: none"> ■ When deleting an instance of a standard NetSuite record type, set this value by using the record.Type enum. ■ When deleting an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	2015.2

Parameter	Type	Required / Optional	Description	Since
options.id	number string	required	The internal ID of the record instance to be deleted.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3
4 // Delete a sales order.
5
6 var salesOrderRecord = record.delete({
7     type: record.Type.SALES_ORDER,
8     id: 88,
9 });
10
11
12 // Delete an instance of a custom record type with the ID customrecord_feature.
13
14 var featureRecord = record.delete({
15     type: 'customrecord_feature',
16     id: 3,
17 });
18 ...
19 // Add additional code.

```

record.delete.promise(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Deletes a record asynchronously.
	<p>i Note: The parameters and errors thrown for this method are the same as those for record.delete(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Supported Script Types	Client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	Transaction records: 20 units Custom records: 4 units All other records: 10 units

Module	N/record Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	<p>The record type.</p> <p>Use the following guidelines:</p> <ul style="list-style-type: none"> ■ When deleting an instance of a standard NetSuite record type, set this value by using the record.Type enum. ■ When deleting an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	2015.2
options.id	number string	required	The internal ID of the record instance to be deleted.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 Important:	The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see Promise Object .
---	---

```

1 // Add additional code.
2
3 // Delete an instance of a standard NetSuite record
4 ...
5
6 function deleteRecord() {
7
8     var deleteRecordPromise = record.delete.promise({
9         type: record.Type.PHONE_CALL,
10        id: 109
11    });
12
13 // To delete an instance of a custom record type, use
14 // the string ID in the type field. For example:
15 // type:'customrecord_feature'
16
17 deleteRecordPromise.then(function() {
18
19     log.debug({
20         title: 'Success',
21         details: 'Record successfully deleted'
22     });
23
24 // Add any other needed code that should execute

```

```

25 // after the record is deleted.
26
27
28 }, function(e) {
29   log.error({
30     title: 'Unable to delete record',
31     details: e.name
32   });
33 });
34 }
35 ...
36 // Add additional code
37

```

record.detach(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Detaches a record from another record.
	 Note: For the promise version of this method, see record.detach.promise(options) . Note that promises are only supported in client scripts.
Returns	void
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/record Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.record	record.Record	required	The record to be detached.	2015.2
options.record.type	string	required	The type of record to be detached. Set this value using the record.Type enum.	2015.2
options.record.id	number string	required	The ID of the record to be detached.	2015.2
options.from	record.Record	required	The destination record that options.record should be detached from.	2015.2
options.from.type	string	required	The type of the destination. Set this value using the record.Type enum.	2015.2

Parameter	Type	Required / Optional	Description	Since
options.from.id	number string	required	The ID of the destination.	2015.2
options.attributes	Object	optional	Name-value pairs containing default values of fields in the new record. By default, this value is null.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 record.detach({
4     record: {
5         type: 'file',
6         id:'200'
7     },
8     from: {
9         type: 'customer',
10        id:'90'
11    }
12 })
13 ...
14 // Add additional code.

```

record.detach.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Detaches a record from another record asynchronously.
	 Note: The parameters and errors thrown for this method are the same as those for record.detach(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/record Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.record	record.Record	required	The record to be detached.	2015.2
options.record.type	string	required	The type of record to be detached. Set this value using the record.Type enum.	2015.2
options.record.id	number string	required	The ID of the record to be detached.	2015.2
options.from	record.Record	required	The destination record that options.record should be detached from.	2015.2
options.from.type	string	required	The type of the destination. Set this value using the record.Type enum.	2015.2
options.from.id	number string	required	The ID of the destination.	2015.2
options.attributes	Object	optional	Name-value pairs containing default values of fields in the new record. By default, this value is null.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code.
2 ...
3 function detachRecord() {
4
5     var detachRecordPromise = record.detach.promise({
6         record: {
7             type: record.Type.CONTACT,
8             id: '98'
9         },
10        from: {
11            type: record.Type.OPPORTUNITY,
12            id: '16'
13        }
14    });
15
16    detachRecordPromise.then(function() {
17
18        // Add any other needed logic that shouldn't execute until
19        // after the contact record is detached from the opportunity.
20    });

```

```

21 |         log.debug({
22 |             title: 'Record updated',
23 |             details: 'Contact record detached'
24 |         });
25 |
26 |     }, function(e) {
27 |         log.error({
28 |             title: e.name,
29 |             details: e.message
30 |         });
31 |     });
32 | }
33 | ...
34 | //Add additional code.

```

record.load(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Loads an existing record.
	<p>i Note: For the promise version of this method, see record.load.promise(options). Note that promises are only supported in client scripts. Make sure to save the record before loading it.</p>
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Transaction records: 10 units Custom records: 2 units All other records: 5 units
Module	N/record Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	<p>The record type. Use the following guidelines:</p> <ul style="list-style-type: none"> ■ When loading an instance of a standard NetSuite record type, set this value by using the record.Type enum. ■ When loading an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	2015.2

Parameter	Type	Required / Optional	Description	Since
options.id	number	required	The internal ID of the existing record instance in NetSuite. The internal ID of the record is displayed on the list page for the record type.	
options.isDynamic	boolean true false	optional	<p>Determines whether the record is loaded in dynamic mode.</p> <ul style="list-style-type: none"> ■ If set to true, the record is loaded in dynamic mode. ■ If set to false, the record is loaded in standard mode. <p>By default, this value is false.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: For additional information on standard and dynamic mode, see record.Record and SuiteScript 2.0 – Standard and Dynamic Modes. </div>	2015.2
options.defaultValues	Object	optional	<p>Name-value pairs containing default values of fields in the new record.</p> <p>By default, this value is null.</p> <p>For a list of available record default values, see N/record Default Values in the NetSuite Help Center.</p>	

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3
4 // Load a sales order.
5
6 var objRecord = record.load({
7   type: record.Type.SALES_ORDER,
8   id: 157,
9   isDynamic: true,
10 });
11
12 // Load an instance of a custom record type with the ID customrecord_feature.
13
14 var newFeatureRecord = record.load({
15   type: 'customrecord_feature',
16   id: 1,
17   isDynamic: true
18 });
19 ...
20 // Add additional code.

```

record.load.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Loads an existing record asynchronously. i Note: The parameters and errors thrown for this method are the same as those for record.load(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Supported Script Types	Client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	Transaction records: 10 units Custom records: 2 units All other records: 5 units
Module	N/record Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	<p>The record type.</p> <p>Use the following guidelines:</p> <ul style="list-style-type: none"> ■ When loading an instance of a standard NetSuite record type, set this value by using the record.Type enum. ■ When loading an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	2015.2
options.id	number	required	<p>The internal ID of the existing record instance in NetSuite. The internal ID of the record is displayed on the list page for the record type.</p>	
options.isDynamic	boolean true false	optional	<p>Determines whether the record is loaded in dynamic mode.</p> <ul style="list-style-type: none"> ■ If set to true, the record is loaded in dynamic mode. ■ If set to false, the record is loaded in standard mode. <p>By default, this value is false.</p>	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p> Note: For additional information on standard and dynamic mode, see record.Record and SuiteScript 2.0 – Standard and Dynamic Modes.</p>	
options.defaultValues	Object	optional	<p>Name-value pairs containing default values of fields in the new record.</p> <p>By default, this value is empty.</p>	

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1  function loadRecord() {
2
3      // Load an instance of a standard NetSuite record
4      // type.
5
6      var loadRecordPromise = record.load.promise({
7          type: record.Type.PHONE_CALL,
8          id: 712
9      });
10
11     // Note: To load an instance of a custom record type,
12     // use the record type's string ID. For example:
13     // type: 'customrecord_feature'
14
15     loadRecordPromise.then(function(objRecord) {
16         objRecord.setValue({
17             fieldId: 'message',
18             value: 'We will start the call with a retrospective.'
19         });
20
21         var recordId = objRecord.save();
22
23         // Add any other needed logic that shouldn't execute
24         // until after the record is instantiated.
25
26         log.debug({
27             title: 'Record updated',
28             details: 'Updated record ID: ' + recordId
29         });
30
31     }, function(e) {
32         log.error({
33             title: 'Unable to load record',
34             details: e.name
35         });
36     });
37 }
38 ...
39 // Add additional code.

```

record.submitFields(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Updates and submits one or more body fields on an existing record in NetSuite, and returns the internal ID of the parent record.</p> <p>When you use this method, you do not need to load or submit the parent record.</p> <p>You can use this method to edit and submit the following:</p> <ul style="list-style-type: none"> ■ Standard body fields that support inline editing (direct list editing). For more information, see the help topic Using Inline Editing. ■ Custom body fields that support inline editing. <p>You cannot use this method to edit and submit the following:</p> <ul style="list-style-type: none"> ■ Select fields ■ Sublist line item fields ■ Subrecord fields (for example, address fields) <p> Note: For the promise version of this method, see record.submitFields.promise(options). Note that promises are only supported in client scripts.</p>
Returns	The internal ID of the parent record.
Supported Script Types	<p>Client and server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	<p>Transaction records: 10 units</p> <p>Custom records: 2 units</p> <p>All other records: 5 units</p>
Module	N/record Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	<p>The record type.</p> <p>Use the following guidelines:</p> <ul style="list-style-type: none"> ■ When working with an instance of a standard NetSuite record type, set this value by using the record.Type enum. ■ When working with an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	2015.2

Parameter	Type	Required / Optional	Description	Since
options.id	number string	required	The internal ID of the existing record instance in NetSuite.	2015.2
options.values	Object	required	<p>The ID-value pairs for each field you want to edit and submit.</p> <p>The value type must correspond to the field type being set. For example:</p> <ul style="list-style-type: none"> ■ Text, Radio, Select and Multi-Select fields accept string values. ■ Checkbox fields accept Boolean values. ■ Date and DateTime fields accept Date values. ■ Integer, Float, Currency and Percent fields accept number values. 	2015.2
options.options	Object	optional	Additional options to set for the record.	2015.2
options.options.enablesourcing	boolean true false	optional	<p>Indicates whether to enable sourcing during the record update.</p> <p>By default, this value is true.</p>	2015.2
options.options.ignoreMandatoryFields	boolean true false	optional	<p>Indicates whether to ignore mandatory fields during record submission.</p> <p>By default, this value is false.</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3
4 // Submit a new value for a sales order's memo field.
5
6 var id = record.submitFields({
7   type: record.Type.SALES_ORDER,
8   id: 1,
9   values: {
10     memo: 'ABC'
11   },
12   options: {
13     enableSourcing: false,
14     ignoreMandatoryFields : true
15   }
16 });
17
18
19 // Submit a new value for a field on an instance of the 'customrecord_book' custom record type.
20

```

```

21 | var otherId = record.submitFields({
22 |   type: 'customrecord_book',
23 |   id: '4',
24 |   values: {
25 |     'custrecord_rating': '2'
26 |   }
27 | });
28 |
29 | ...
30 | // Add additional code.

```

record.submitFields.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Updates and submits one or more body fields asynchronously on an existing record in NetSuite, and returns the internal ID of the parent record.</p> <p>When you use this method, you do not need to load or submit the parent record.</p> <p>You can use this method to edit and submit the following:</p> <ul style="list-style-type: none"> ■ Standard body fields that support inline editing (direct list editing). For more information, see the help topic Using Inline Editing. ■ Custom body fields that support inline editing. <p>You cannot use this method to edit and submit the following:</p> <ul style="list-style-type: none"> ■ Select fields ■ Sublist line item fields ■ Subrecord fields (for example, address fields) <p> Note: The parameters and errors thrown for this method are the same as those for record.submitFields(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Supported Script Types	<p>Client-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Client Script Type.</p>
Governance	<p>Transaction records: 10 units</p> <p>Custom records: 2 units</p> <p>All other records: 5 units</p>
Module	N/record Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	string	required	The record type.	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p>Use the following guidelines:</p> <ul style="list-style-type: none"> ■ When working with an instance of a standard NetSuite record type, set this value by using the <code>record.Type</code> enum. ■ When working with an instance of a custom record type, set this value by using the custom record type's string ID. For help finding this ID, see the help topic Custom Record. 	
options.id	number string	required	The internal ID of the existing record instance in NetSuite.	2015.2
options.values	Object	required	The ID-value pairs for each field you want to edit and submit.	2015.2
options.options	Object	optional	Additional options to set for the record.	2015.2
options.options.enablesourcing	boolean true false	optional	<p>Indicates whether to enable sourcing during the record update. By default, this value is true.</p>	2015.2
options.options.ignoreMandatoryFields	boolean true false	optional	<p>Indicates whether to ignore mandatory fields during record submission. By default, this value is false.</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 function submitFields() {
4
5     var submitFieldsPromise = record.submitFields.promise({
6         type: record.Type.PHONE_CALL,
7         id: 171,
8         values: {
9             title: 'Sprint 3 planning'
10        },
11    });
12
13    submitFieldsPromise.then(function(recordId) {
14
15        // Add any needed logic that shouldn't execute until
16        // after the new value is submitted.
17
18        log.debug({
19            title: 'Record updated',
20            details: 'Id of updated record: ' + recordId
21        });

```

```

22     }, function(e) {
23       log.error({
24         title: e.name,
25         details: e.message
26       });
27     });
28   });
29 }
30 ...
31 // Add additional code

```

record.transform(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Transforms a record from one type into another, using data from an existing record. You can use this method to automate order processing, creating item fulfillment transactions and invoices off of orders. For a list of supported transformations, see Supported Transformation Types .
Returns	record.Record
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	Transaction records: 10 units Custom records: 2 units All other record types: 5 units
Module	N/record Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fromType	string	required	The record type of the existing record instance being transformed. This value sets the Record.type property for the record. This property is read-only and cannot be changed after the record is loaded. Set this value using the record.Type .	2015.2
options.fromId	number	required	The internal ID of the existing record instance being transformed.	2015.2

Parameter	Type	Required / Optional	Description	Since
options.toType	string	required	The record type of the record returned when the transformation is complete.	2015.2
options.isDynamic	boolean true false	optional	<p>Determines whether the new record is created in dynamic mode.</p> <ul style="list-style-type: none"> ■ If set to true, the new record is created in dynamic mode. ■ If set to false, the new record is created in standard mode. <p>By default, this value is false.</p> <p>Note: For additional information on standard and dynamic mode, see record.Record and SuiteScript 2.0 – Standard and Dynamic Modes.</p>	2015.2
options.defaultValue	Object	optional	<p>Name-value pairs containing default values of fields in the new record.</p> <p>By default, this value is null.</p> <p>For a list of available record default values, see N/record Default Values in the NetSuite Help Center.</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.transform({
4   fromType: record.Type.CUSTOMER,
5   fromId: 107,
6   toType: record.Type.SALES_ORDER,
7   isDynamic: true,
8 });
9 ...
10 // Add additional code.

```

```

1 // Add additional code.
2 ...
3 record.transform({
4   fromType:'salesorder',
5   fromId: 6,
6   toType: 'invoice',
7   defaultValues: {
8     billdate: '01/01/2019' } });
9 ...

```

```
10 | // Add additional code.
```

Supported Transformation Types

Original Record Name	Original Record Type	Transformed Record Name	Transformed Record Type
Build/Assembly	assemblyitem	Assembly Build	assemblybuild
Assembly Build	assemblybuild	Assembly Unbuild	assemblyunbuild
Cash Sale	cashsale	Cash Sale	cashrefund
Customer	customer	Cash Sale	cashsale
Customer	customer	Customer Payment	customerpayment
		<p>Note: When you use this transformation, do not use <code>Record.setValue(options)</code> to set the value of the customer field on the resulting Customer Payment record. This field is populated automatically during transformation, and you cannot specify a value for this field after the transformation is complete.</p>	
Customer	customer	Quote	estimate
Customer	customer	Invoice	invoice
Customer	customer	Opportunity	opportunity
Customer	customer	Sales Order	salesorder
Employee	employee	Expense Report	expensereport
Employee	employee	Time	timebill
Quote	estimate	Cash Sale	cashsale
Quote	estimate	Invoice	invoice
Quote	estimate	Sales Order	salesorder
Intercompany Transfer Order	intercompanytransferorder	Item Fulfillment	itemfulfillment
Intercompany Transfer Order	intercompanytransferorder	Item Receipt	itemreceipt
Invoice	invoice	Credit Memo	creditmemo
Invoice	invoice	Customer Payment	customerpayment
Invoice	invoice	Return Authorization	returnauthorization
Lead	lead	Opportunity	opportunity
Opportunity	opportunity	Cash Sale	cashsale
Opportunity	opportunity	Quote	estimate
Opportunity	opportunity	Invoice	invoice
Opportunity	opportunity	Sales Order	salesorder
Prospect	prospect	Quote	estimate
Prospect	prospect	Opportunity	opportunity

Original Record Name	Original Record Type	Transformed Record Name	Transformed Record Type
Prospect	prospect	Sales Order	salesorder
Purchase Order	purchaseorder	Item Receipt	itemreceipt
Purchase Order	purchaseorder	Vendor Bill	vendorbill
Purchase Order	purchaseorder	Vendor Return Authorization	vendorreturnauthorization
Return Authorization	returnauthorization	Cash Refund	cashrefund
Return Authorization	returnauthorization	Credit Memo	creditmemo
Return Authorization	returnauthorization	Item Receipt	itemreceipt
Return Authorization	returnauthorization	Revenue Commitment Reversal	revenuecommitmentreversal
<p> Note: The return authorization must be approved and received for this transformation to work.</p>			
Sales Order	salesorder	Cash Sale	cashsale
Sales Order	salesorder	Invoice	invoice
Sales Order	salesorder	Item Fulfillment	itemfulfillment
Sales Order	salesorder	Return Authorization	returnauthorization
Sales Order	salesorder	Revenue Commitment	revenuecommitment
Transfer Order	transferorder	Item Fulfillment	itemfulfillment
Transfer Order	transferorder	Item Receipt	itemreceipt
Vendor	vendor	Purchase Order	purchaseorder
Vendor	vendor	Vendor Bill	vendorbill
Vendor Bill	vendorbill	Vendor Credit	vendorcredit
Vendor Bill	vendorbill	Vendor Payment	vendorpayment
Vendor Bill	vendorbill	Vendor Return Authorization	vendorreturnauthorization
Vendor Return Authorization	vendorreturnauthorization	Item Fulfillment	itemfulfillment
Vendor Return Authorization	vendorreturnauthorization	Vendor Credit	vendorcredit
Work Order	workorder	Assembly Build	assemblybuild
Work Order	workorder	Work Order Close	workorderclose
Work Order	workorder	Work Order Completion	workordercompletion
Work Order	workorder	Work Order Issue	workorderissue

record.transform.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Transforms a record from one type into another asynchronously, using data from an existing record.</p> <p>You can use this method to automate order processing, creating item fulfillment transactions and invoices off of orders.</p> <p>For a list of supported transformations, see Supported Transformation Types.</p>
	<p>Note: The parameters and errors thrown for this method are the same as those for record.transform(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Supported Script Types	<p>Client-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Client Script Type.</p>
Governance	<p>Transaction records: 10 units</p> <p>Custom records: 2 units</p> <p>All other record types: 5 units</p>
Module	N/record Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.fromType	string	required	<p>The record type of the existing record instance being transformed.</p> <p>This value sets the Record.type property for the record. This property is read-only and cannot be changed after the record is loaded.</p> <p>Set this value using the record.Type.</p>	2015.2
options.fromId	number	required	The internal ID of the existing record instance being transformed.	2015.2
options.toType	string	required	The record type of the record returned when the transformation is complete.	2015.2
options.isDynamic	boolean true false	optional	<p>Determines whether the new record is created in dynamic mode.</p> <ul style="list-style-type: none"> ■ If set to true, the new record is created in dynamic mode. 	2015.2

Parameter	Type	Required / Optional	Description	Since
			<ul style="list-style-type: none"> ■ If set to false, the new record is created in standard mode. <p>By default, this value is false.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: For additional information on standard and dynamic mode, see record.Record and SuiteScript 2.0 – Standard and Dynamic Modes. </div>	
options.defaultValues	Object	optional	<p>Name-value pairs containing default values of fields in the new record.</p> <p>By default, this value is null.</p>	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required argument is missing or undefined.

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code.
2 ...
3 function transformRecord() {
4
5     var transformRecordPromise = record.transform.promise({
6         fromType: record.Type.estimate,
7         fromId: 25,
8         toType: record.Type.SALES_ORDER,
9         isDynamic: true,
10    });
11
12    transformRecordPromise.then(function(recordObject) {
13
14        var recordId = recordObject.save();
15
16        // Add any other needed logic that shouldn't execute until
17        // after the record is transformed.
18
19        log.debug({
20            title: 'Record saved',
21            details: 'Id of new record: ' + recordId
22        });
23
24    }, function(e) {
25
26        log.error({
27            title: e.name,
28            details: e.message
29        });
30    });
31
32 ...
33 // Add additional code.

```

record.Type

Note:	The content in this help topic pertains to SuiteScript 2.0.
Note:	JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Enumeration that holds the string values for supported record types. This enum is used to set the value of the Record.type property in cases where you are working with an instance of a standard NetSuite record type. (If you are working with an instance of a custom record type, you set the Record.type property by using the custom record type's string ID. For more help finding this ID, see the help topic Custom Record .)
Supported Script Types	Client and server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/record Module
Since	2015.2

Values

■ ACCOUNT	■ EMAIL_TEMPLATE	■ PRICE_BOOK
■ ACCOUNTING_BOOK	■ EMPLOYEE	■ PRICE_LEVEL
■ ACCOUNTING_CONTEXT	■ EMPLOYEE_CHANGE_REQUEST	■ PRICE_PLAN
■ ACCOUNTING_PERIOD	■ EMPLOYEE_CHANGE_TYPE	■ PRICING_GROUP
■ ADV_INTER_COMPANY_JOURNAL_ENTRY	■ ENTITY_ACCOUNT_MAPPING	■ PROJECT_EXPENSE_TYPE
■ ALLOCATION_SCHEDULE	■ ESTIMATE	■ PROJECT_TASK
■ AMORTIZATION_SCHEDULE	■ EXPENSE_AMORTIZATION_EVENT	■ PROJECT_TEMPLATE
■ AMORTIZATION_TEMPLATE	■ EXPENSE_CATEGORY	■ PROMOTION_CODE
■ AS_CHARGED_PROJECT_REVENUE_RULE	■ EXPENSE_PLAN	■ PROSPECT
■ ASSEMBLY_BUILD	■ EXPENSE_REPORT	■ PURCHASE_CONTRACT
■ ASSEMBLY_ITEM	■ FAIR_VALUE_PRICE	■ PURCHASE_ORDER
■ ASSEMBLY_UNBUILD	■ FINANCIAL_INSTITUTION	■ PURCHASE_REQUSITION
■ BALANCE_TRX_BY_SEGMENTS	■ FIXED_AMOUNT_PROJECT_REVENUE_RULE	■ REALLOCATE_ITEM
■ BILLING_ACCOUNT	■ FOLDER	■ RECEIVE_INBOUND_SHIPMENT
■ BILLING_CLASS	■ FULFILLMENT_REQUEST	■ RESOURCE_ALLOCATION
■ BILLING_RATE_CARD	■ GENERAL_TOKEN	■ RESTLET
■ BILLING_REVENUE_EVENT	■ GENERIC_RESOURCE	■ RETURN_AUTHORIZATION
■ BILLING_SCHEDULE	■ GIFT_CERTIFICATE	■ REVENUE_ARRANGEMENT
■ BIN	■ GIFT_CERTIFICATE_ITEM	■ REVENUE_COMMITMENT
■ BIN_TRANSFER	■ GLOBAL_ACCOUNT_MAPPING	■ REVENUE_COMMITMENT_REVERSAL
■ BIN_WORKSHEET	■ GLOBAL_INVENTORY_RELATIONSHIP	■ REVENUE_PLAN
■ BLANKET_PURCHASE_ORDER	■ GL_NUMBERING_SEQUENCE	■ REV_REC_SCHEDULE
■ BOM	■ GOAL	■ REV_REC_TEMPLATE
■ BOM_REVISION	■ INBOUND_SHIPMENT	■ SALES_ORDER
■ BONUS	■ INTERCOMP_ALLOCATION_SCHEDULE	■ SALES_ROLE
	■ INTER_COMPANY_JOURNAL_ENTRY	■ SALES_TAX_ITEM

■ BONUS_TYPE	■ INTER_COMPANY_TRANSFER_ORDER	■ SCHEDULED_SCRIPT
■ BUDGET_EXCHANGE_RATE	■ INVENTORY_ADJUSTMENT	■ SCHEDULED_SCRIPT_INSTANCE
■ BULK_OWNERSHIP_TRANSFER	■ INVENTORY_COST_REVALUATION	■ SCRIPT_DEPLOYMENT
■ BUNDLE_INSTALLATION_SCRIPT	■ INVENTORY_COUNT	■ SERIALIZED_ASSEMBLY_ITEM
■ CALENDAR_EVENT	■ INVENTORY_DETAIL	■ SERIALIZED_INVENTORY_ITEM
■ CAMPAIGN	■ INVENTORY_ITEM	■ SERVICE_ITEM
■ CAMPAIGN_RESPONSE	■ INVENTORY_NUMBER	■ SHIP_ITEM
■ CAMPAIGN_TEMPLATE	■ INVENTORY_STATUS	■ SOLUTION
■ CASH_REFUND	■ INVENTORY_STATUS_CHANGE	■ STATISTICAL_JOURNAL_ENTRY
■ CASH_SALE	■ INVENTORY_TRANSFER	■ STORE_PICKUP_FULFILLMENT
■ CHARGE	■ INVOICE	■ SUBSCRIPTION
■ CHARGE_RULE	■ ISSUE	■ SUBSCRIPTION_CHANGE_ORDER
■ CHECK	■ ISSUE_PRODUCT	■ SUBSCRIPTION_LINE
■ CLASSIFICATION	■ ISSUE_PRODUCT_VERSION	■ SUBSCRIPTION_PLAN
■ CLIENT_SCRIPT	■ ITEM_ACCOUNT_MAPPING	■ SUBSIDIARY
■ CMS_CONTENT	■ ITEM_COLLECTION	■ SUBSIDIARY_SETTINGS
■ CMS_CONTENT_TYPE	■ ITEM_COLLECTION_ITEM_MAP	■ SUBTOTAL_ITEM
■ CMS_PAGE	■ ITEM_DEMAND_PLAN	■ SUITELET
■ COMMERCE_CATEGORY	■ ITEM_FULFILLMENT	■ SUPPLY_CHAIN_SNAPSHOT
■ COMPETITOR	■ ITEM_GROUP	■ SUPPLY_CHAIN_SNAPSHOT_SIMULATION
■ CONSOLIDATED_EXCHANGE_RATE	■ ITEM_LOCATION_CONFIGURATION	■ SUPPORT_CASE
■ CONTACT	■ ITEM_PROCESS_FAMILY	■ TASK
■ CONTACT_CATEGORY	■ ITEM_PROCESS_GROUP	■ TAX_ACCT
■ CONTACT_ROLE	■ ITEM_RECEIPT	■ TAX_GROUP
■ COST_CATEGORY	■ ITEM_REVISION	■ TAX_PERIOD
■ COUPON_CODE	■ ITEM_SUPPLY_PLAN	■ TAX_TYPE
■ CREDIT_CARD_CHARGE	■ JOB	■ TERM
■ CREDIT_CARD_REFUND	■ JOB_STATUS	■ TIME_BILL
■ CREDIT_MEMO	■ JOB_TYPE	■ TIME_ENTRY
■ CURRENCY	■ JOURNAL_ENTRY	■ TIME_OFF_CHANGE
■ CUSTOM_PURCHASE	■ KIT_ITEM	■ TIME_OFF_PLAN
■ CUSTOMER	■ LABOR_BASED_PROJECT_REVENUE_RULE	■ TIME_OFF_REQUEST
■ CUSTOMER_CATEGORY	■ LEAD	■ TIME_OFF_RULE
■ CUSTOMER_DEPOSIT	■ LOCATION	■ TIME_OFF_TYPE
■ CUSTOMER_MESSAGE	■ LOT_NUMBERED_ASSEMBLY_ITEM	■ TIME_SHEET
■ CUSTOMER_PAYMENT	■ LOT_NUMBERED_INVENTORY_ITEM	■ TOPIC
■ CUSTOMER_PAYMENT_AUTHORIZATION	■ MANUFACTURING_COST_TEMPLATE	■ TRANSACTION
■ CUSTOMER_REFUND	■ MANUFACTURING_OPERATION_TASK	■ TRANSFER_ORDER
■ CUSTOMER_STATUS	■ MANUFACTURING_ROUTING	■ UNITS_TYPE
■ CUSTOMER_SUBSIDIARY_RELATIONSHIP	■ MAP_REDUCE_SCRIPT	■ UNLOCKED_TIME_PERIOD
■ CUSTOM_RECORD	■ MARKUP_ITEM	■ USAGE
■ CUSTOM_TRANSACTION	■ MASSUPDATE_SCRIPT	■ USEREVENT_SCRIPT
■ DEPARTMENT	■ MEM_DOC	■ VENDOR
■ DEPOSIT	■ MERCHANTISE_HIERARCHY_LEVEL	■ VENDOR_BILL
■ DEPOSIT_APPLICATION	■ MERCHANTISE_HIERARCHY_NODE	■ VENDOR_CATEGORY
■ DESCRIPTION_ITEM	■ MERCHANTISE_HIERARCHY_VERSION	■ VENDOR_CREDIT
	■ MESSAGE	■ VENDOR_PAYMENT

<ul style="list-style-type: none"> ■ DISCOUNT_ITEM ■ DOWNLOAD_ITEM 	<ul style="list-style-type: none"> ■ MFG_PLANNED_TIME ■ NEXUS ■ NON_INVENTORY_ITEM ■ NOTE ■ NOTE_TYPE ■ OPPORTUNITY ■ ORDER_SCHEDULE ■ OTHER_CHARGE_ITEM ■ OTHER_NAME ■ OTHER_NAME_CATEGORY ■ PARTNER ■ PARTNER_CATEGORY ■ PARTNER_COMMISSION_PLAN ■ PAYCHECK ■ PAYCHECK_JOURNAL ■ PAYMENT_CARD ■ PAYMENT_CARD_TOKEN ■ PAYMENT_ITEM ■ PAYMENT_METHOD ■ PAYROLL_ITEM ■ PCT_COMPLETE_PROJECT_REVENUE_RULE ■ PERFORMANCE_METRIC ■ PERFORMANCE REVIEW ■ PERFORMANCE REVIEW_SCHEDULE ■ PERIOD_END_JOURNAL ■ PHONE_CALL ■ PICK_STRATEGY ■ PICK_TASK ■ PORTLET 	<ul style="list-style-type: none"> ■ VENDOR_PREPAYMENT ■ VENDOR_PREPAYMENT_APPLICATION ■ VENDOR_RETURN_AUTHORIZATION ■ VENDOR_SUBSIDIARY_RELATIONSHIP ■ WAVE ■ WBS ■ WEBSITE ■ WORKFLOW_ACTION_SCRIPT ■ WORK_ORDER ■ WORK_ORDER_CLOSE ■ WORK_ORDER_COMPLETION ■ WORK_ORDER_ISSUE ■ WORKPLACE ■ ZONE
--	--	---

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/record Module Script Samples](#).

```

1 // Add additional code.
2 ...
3 var objRecord = record.delete({
4   type: record.Type.SALES_ORDER,
5   id: 128
6 });
7 ...
8 // Add additional code.

```

N/recordContext Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Use the N/recordContext Module to get all the available context types of the record, such as LOCALIZATION. The LOCALIZATION context type indicates which country a script is using for execution.

N/recordContext Module members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	recordContext. RecordContext	Object	Client and server-side scripts	This is an object of name/value pair.
Method	recordContext.getContext (options)	Object	Client and server-side scripts	Returns the record context object for a record.
Enum	recordContext.ContextType	enum	Client and server-side scripts	Holds the values for the context type.

N/recordContext Module Sample



Note: This sample script uses the require function so you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (a script you attach to a script record and deploy). For more information, see the help topic [SuiteScript 2.0 Script Basics](#) SuiteScript 2.0 Script Basics and [SuiteScript 2.0 Script Types](#) SuiteScript 2.0 Script Types.

```

1 /**
2  * @NApiVersion 2.x
3 */
4 // This example calls the getContext() API with options recordType, recordId and record.
5 require(['N/record', 'N/recordContext'],
6   function(record, recordContext) {
7     // Create record
8     var employee = record.create({
9       type : record.Type.EMPLOYEE,
10      isDynamic: true
11    })
12    employee.setValue('subsidiary', 2); // Setup CA subsidiary
13    employee.setValue('entityid', 'test_emp_' + Date.now())
14    employeeId = employee.save()
15
16 // getContext() with options recordType, recordId and contextTypes
17 var employeeContext = recordContext.getContext({
18   recordType : record.Type.EMPLOYEE,
19   recordId: employeeId,
20   contextTypes: [recordContext.ContextType.LOCALIZATION]
21 })
22 log.debug(employeeContext); // log debug {"localization":["CA"]}
23
24 employee.setValue('subsidiary', 3); // Setup AU subsidiary
25 // getContext() with options record and contextTypes
26 var employeeContext = recordContext.getContext({
27   record : employee,
28   contextTypes: [recordContext.ContextType.LOCALIZATION]
```

```

29     })
30     log.debug(employeeContext); //log debug {"localization":["AU"]}
31
32     // Delete record
33     record.delete({
34         type : record.Type.EMPLOYEE,
35         id: employeeId
36     })
37 });

```

recordContext.RecordContext

Object Description	This is an object of name/value pair. Each name is the name of the context type. Each value is the record context. More values for one context type can be returned in an array.
Supported Script Types	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/recordContext Module
Methods and Properties	N/recordContext Module members
Since	2020.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/recordContext Module Sample](#).

```

1 // Add additional code
2
3 {"localization":["CA"]}
4 ...
5 // Add additional code

```

recordContext.getContext(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the record context object for a given record.
Returns	Object
Supported Script Types	Client and server-side scripts
Governance	10 units
Module	N/recordContext Module
Since	2020.2

Parameters



Note: The options parameter is a JavaScript object.



Important: For records that are not loaded, the record is defined by record type and ID. For records that are loaded, the record is defined by the record object.

Parameter	Type	Required / Optional	Description	Since
options.recordType	number	required	The record type.	2020.2
options.recordID	string	required	The record ID.	2020.2
options.record	string	required	The record object.	2020.2
options.contextTypes	Array <string>	optional	The available context types.	2020.2

Errors

Error Code	Thrown If
MUTUALLY_EXCLUSIVE_ARGUMENT	The record is present alongside a record ID or record type.
MISSING_REQD_ARGUMENT	Any of the record ID or record type parameter is missing.
SSS_INVALID_TYPE_ARG	The parameter type is wrong.
UNKNOWN_CONTEXT_TYPE	The context type selection is unknown.
INVALID_UNSUPRTD_RCRD_TYP	The record type selection is invalid or not supported.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/recordContext Module Sample](#).

```

1 // Add additional code
2 // Sample code on getting context for records that are not loaded
3 ...
4 var employeeContext = recordContext.getContext({
5   recordType: record.Type.EMPLOYEE,
6   recordId: employeeId
7   contextTypes: [recordContext.ContextType.LOCALIZATION]
8 ...
9 // Add additional code

```

```

1 // Add additional code
2 // Sample code on getting context for records that are loaded
3 ...
4 var lrc = recordContext.getContext({
5   record: recordObject,
6   contextTypes: [recordContext.ContextType.LOCALIZATION]
7 ...
8 // Add additional code

```

recordContext.ContextType

Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the value for the context type.
Type	enum
Module	N/recordContext Module
Sibling Module Members	N/recordContext Module members
Since	2020.2

Values

LOCALIZATION

The localization is indicated by country codes as defined by ISO 3166-2.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/recordContext Module Sample](#).

```

1 // Add additional code
2 ...
3 // getContext() with options record and contextTypes
4 var employeeContext = recordContext.getContext({
5
6     record: employee,
7
8     contextTypes: [recordContext.ContextType.LOCALIZATION]
9
10 })
11 ...
12 // Add additional code

```

N/redirect Module

Note: The content in this help topic pertains to SuiteScript 2.0.

Use the redirect module to customize navigation within NetSuite by setting up a redirect URL that resolves to a NetSuite resource or external URL. You can redirect users to one of the following:

- URL
- Suitelet
- Record
- Task link
- Saved search
- Unsaved search



Note: Suitelets, beforeLoad user events, and synchronous afterSubmit user events are supported. This module does not support beforeSubmit and asynchronous afterSubmit user events.

- [N/redirect Module Members](#)
- [N/redirect Module Script Samples](#)

N/redirect Module Members

Member Type	Name	Return Type	Supported Script Types	Description
Method	redirect.redirect(options)	void	Suitelets, beforeLoad user events, and synchronous afterSubmit user events	Redirects to the URL of a Suitelet that is available externally (available without login).
	redirect.toRecord(options)	void	Suitelets, beforeLoad user events, and synchronous afterSubmit user events	Redirects to a NetSuite record.
	redirect.toRecordTransform(options)	void	workflow action scripts	Redirects to a standard or custom transaction instance.
	redirect.toSavedSearch(options)	void	afterSubmit user events	Redirects to a saved search.
	redirect.toSavedSearchResult(options)	void	afterSubmit user events	Redirects to a saved search result.
	redirect.toSearch(options)	void	afterSubmit user events	Redirects to search.
	redirect.toSearchResult(options)	void	afterSubmit user events	Redirects to search results.
	redirect.toSuitelet(options)	void	Suitelets, beforeLoad user events, and synchronous afterSubmit user events	Redirects to a Suitelet.
	redirect.toTaskLink(options)	void	Suitelets, beforeLoad user events, and synchronous afterSubmit user events	Redirects to a tasklink.

N/redirect Module Script Samples

The following script samples demonstrate how to use the features of the N/redirect module.

Sample 1: Redirect to a task record



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample sets the redirect URL to a newly created task record. To set a redirect using a record ID, the record must have been previously submitted to NetSuite.

```

1  /**
2  * @NApiVersion 2.x
3 */
4
5  require(['N/record', 'N/redirect'], function(record, redirect) {
6      function redirectToTaskRecord() {
7          var taskTitle = 'New Opportunity';
8          var taskRecord = record.create({
9              type: record.Type.TASK
10         });
11         taskRecord.setValue('title', taskTitle);
12
13         var taskRecordId = taskRecord.save();
14
15         redirect.toRecord({
16             type: record.Type.TASK,
17             id: taskRecordId
18         });
19     }
20
21     redirectToTaskRecord();
22 });

```

redirect.redirect(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to set the redirect to the URL of a Suitelet that is available externally (Suitelets set to Available Without Login on the Script Deployment page).
Returns	Void
Supported Script Types	Suitelets, beforeLoad user events, and synchronous afterSubmit user events For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/redirect Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.url	string	required	The URL of a Suitelet that is available externally  Note: For an external URL, Available without Login must be enabled on the Script Deployment page for the Suitelet.
options.parameters	Object	optional	Contains additional URL parameters as key/value pairs.

Parameter	Type	Required / Optional	Description
			<p>Note: Parameters cannot be arrays. Use JSON.stringify/JSON.parse instead to handle arrays.</p>

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see redirect Module Script Sample.

```

1 //Add additional code
2 ...
3 redirect.redirect({
4   url: '/app/site/hosting/scriptlet.nl?script=130&deploy=1',
5   parameters: {'custparam_test':'helloWorld'}
6 });
7 ...
8 //Add additional code

```

redirect.toRecord(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to set the redirect URL to a specific NetSuite record.
	<p>Note: If you redirect a user to a record, the record must first exist in NetSuite. If you want to redirect a user to a new record, you must first create and submit the record before redirecting them. You must also ensure that any required fields for the new record are populated before submitting the record.</p>
Returns	Void
Supported Script Types	Suitelets, beforeLoad user events, and synchronous afterSubmit user events For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/redirect Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The internal id of the target record.

Parameter	Type	Required / Optional	Description
options.type	string	required	The type of record.
options.isEditMode	boolean true false	optional	Determines whether to return a URL for the record in edit mode or view mode. If set to true, returns the URL to an existing record in edit mode. The default value is false – returns the URL to a record in view mode.
options.parameters	Object	optional	Contains additional URL parameters as key/value pairs.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [redirect Module Script Sample](#).

```

1 //Add additional code
2 ...
3 redirect.toRecord({
4   type : record.Type.TASK,
5   id : taskRecordId,
6   parameters: {'custparam_test':'helloWorld'}
7 });
8 ...
9 //Add additional code

```

redirect.toRecordTransform(options)

ℹ Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Transforms a record to a standard or custom transaction instance. The fromId and fromType parameters can be acquired from the context object provided by the onAction(scriptContext) method of a workflow action script. The toType record type identifier cannot be acquired programmatically and must be entered manually. The new transaction instance opens in edit mode with all possible fields defaulted from the source record. To specify URL parameters or record field values on the redirected page, use the options.parameters parameter. For record field values, use the record.fieldId format. Invalid keys are ignored. The user can edit transaction details before saving. For a list of supported transformation types, see Supported Transformation Types .
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/redirect Module
Sibling Module Members	N/redirect Module Members

Since	2020.1
-------	--------

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.fromId	number	required	The ID of the source record.
options.fromType	string	required	The ID of the source record type.
options.toType	string	required	The ID of the target record type.
options.parameters	Object	optional	Contains additional parameters as key/value pairs.

Errors

Error Code	Error Message	Thrown If
INVALID_RCRD_TYPE	The record type 'type' is invalid.	The record type specified in the toType parameter is invalid.
INVALID_RCRD_TRANSFRM	That type of record transformation is not allowed. Please see the documentation for a list of supported transformation types.	Transformation between the fromType and toType record types is not allowed.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/redirect Module Script Samples](#).

```

1 // Add additional code
2 ...
3 define(['N/redirect'], function (redirect) {
4     return {
5         onAction: function (context)
6         {
7             redirect.toRecordTransform({
8                 fromType: context.newRecord.type,
9                 fromId: context.newRecord.id,
10                toType: 'invoice',
11                parameters: {
12                    'record.memo': 'memo override', // override of memo field on body form
13                    'cf':'92' //std product invoice custom form
14                }
15            });
16        },
17    });
18 ...
20 // Add additional code

```

redirect.toSavedSearch(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to load an existing saved search and redirect to the populated search definition page.
Returns	Void
Supported Script Types	afterSubmit user event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Governance	5 units
Module	N/redirect Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	number	required	<p>Internal ID of the search.</p> <p>The internal ID is available only when the search is either loaded with search.load(options) or after it has been saved with Search.save().</p> <p>Typical values are 55 or 234 or 87, not a value like customsearch_mysearch. Any ID prefixed with customsearch is a script ID, not the internal system ID for a search.</p>

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [redirect Module Script Sample](#).

```

1 //Add additional code
2 ...
3 redirect.toSavedSearch({id: 234});
4 ...
5 //Add additional code

```

redirect.toSavedSearchResult(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to redirect a user to a search results page for an existing saved search.
Returns	Void

Supported Script Types	afterSubmit user event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Governance	5 units
Module	N/redirect Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	number	required	<p>Internal ID of the search.</p> <p>The internal ID is available only when the search is either loaded with search.load(options) or after it has been saved with Search.save().</p> <p>Typical values are 55 or 234 or 87, not a value like customsearch_mysearch. Any ID prefixed with customsearch is a script ID, not the internal system ID for a search.</p>

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [redirect Module Script Sample](#).

```

1 //Add additional code
2 ...
3 redirect.toSavedSearchResult({id: 234});
4 ...
5 //Add additional code

```

redirect.toSearch(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to redirect a user to an on demand search built in SuiteScript. This method loads a search into the session, and then redirects to a URL that loads the search definition page.
Returns	Void
Supported Script Types	afterSubmit user event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Governance	None
Module	N/redirect Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.search	search.Search	required	

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see redirect Module Script Sample.

```

1 var column = ['internalid'];
2 var filter = [{"mainline": "is", "T"}];
3 var ourNewSearch = search.create({
4   id: 'customsearch_test',
5   type: search.Type.SALES_ORDER,
6   title: 'My Generated Search',
7   columns: column,
8   filters: filter
9 });
10 redirect.toSearch({
11   search: ourNewSearch
12 });

```

redirect.toSearchResult(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to redirect a user to a search results page. For example, the results from an on demand search created with the N/search Module , or a loaded search that you modified but did not save.
Returns	Void
Supported Script Types	afterSubmit user event scripts For more information, see the help topic SuiteScript 2.0 User Event Script Type .
Governance	None
Module	N/redirect Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.Search	search.Search	required	

redirect.toSuitelet(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to redirect the user to a Suitelet. For more information about Suitelets, see the help topic SuiteScript 2.0 Suitelet Script Type .
	 Note: The redirect happens after the script finishes.
Returns	Void
Supported Script Types	Suitelets, beforeLoad user events, and synchronous afterSubmit user events For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/redirect Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.scriptId	string	required	The script ID for the Suitelet.
options.deploymentId	string	required	The deployment ID for the Suitelet.
options.isExternal	boolean true false	optional	The default value is false – indicates an external Suitelet URL.
options.parameters	Object	optional	Contains additional URL parameters as key/value pairs.
			 Note: Parameters cannot be arrays. Use JSON.stringify/JSON.parse instead to handle arrays.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [redirect Module Script Sample](#).

```

1 //Add additional code
2 ...
3 redirect.toSuitelet({
4  scriptId: 31 ,
5   deploymentId: 1,
6   parameters: {'custparam_test':'helloWorld'}
7 });

```

```

8 ...
9 //Add additional code

```

redirect.toTaskLink(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to redirect a user to a tasklink.
Returns	Void
Supported Script Types	Suitelets, beforeLoad user events, and synchronous afterSubmit user events For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/redirect Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The taskId for a tasklink For a list of supported task IDs, see the help topic Task IDs .
options.parameters	Object	optional	Contains additional URL parameters as key/value pairs.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [redirect Module Script Sample](#).

```

1 //Add additional code
2 ...
3 redirect.toTaskLink({
4   id: 'ADMI_SHIPPING' ,
5   parameters: {'custparam_test':'helloWorld'}
6 });
7 ...
8 //Add additional code

```

N/render Module

Note: The content in this help topic pertains to SuiteScript 2.0.

The render module encapsulates functionality for printing, PDF creation, form creation from templates, and email creation from templates.



Note: Direct manipulation of the print URL is **not** supported.

- N/render Module Members
- EmailMergeResult Object Members
- TemplateRenderer Object Members
- N/render Module Script Sample

N/render Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	render.EmailMergeResult	Object	Server-side scripts	Encapsulates an email merge result
	render.TemplateRenderer	Object	Server-side scripts	Encapsulates a template engine object that produces HTML and PDF printed forms utilizing advanced PDF/HTML template capabilities
Method	render.bom(options)	file.File	Server-side scripts	Creates a PDF or HTML file object containing a bill of materials
	render.create()	render.TemplateRenderer	Server-side scripts	Creates a render.TemplateRenderer object
	render.mergeEmail(options)	render.EmailMergeResult	Server-side scripts	Creates a render.EmailMergeResult object
	render.packingSlip(options)	file.File	Server-side scripts	Creates a PDF or HTML file object containing a packing slip
	render.pickingTicket(options)	file.File	Server-side scripts	Creates a PDF or HTML file object containing a picking ticket
	render.statement(options)	file.File	Server-side scripts	Creates a PDF or HTML file object containing a statement
	render.transaction(options)	file.File	Server-side scripts	Creates a PDF or HTML file object containing a transaction
	render.xmlToPdf(options)	file.File	Server-side scripts	Passes XML to the BFO tag library (which is stored by NetSuite), and returns a PDF file
Enum	render.DataSource	enum	Server-side scripts	Holds the string values for supported data source types
	render.PrintMode	enum	Server-side scripts	Holds the string values for supported print output types

EmailMergeResult Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	EmailMergeResult.body	string (read-only)	Server-side scripts	The body of the email distribution in string format

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	EmailMergeResult.subject	string (read-only)	Server-side scripts	The subject of the email distribution in string format

TemplateRenderer Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	TemplateRenderer.addCustomDataSource(options)	void	Server-side scripts	Adds an XML file or JSON object to an advanced template as a custom data source
	TemplateRenderer.addQuery(options)	void	Server-side scripts	Uses Query as the renderer's data source
	TemplateRenderer.addRecord(options)	void	Server-side scripts	Binds a record to a template variable
	TemplateRenderer.addSearchResults(options)	void	Server-side scripts	Binds a search result to a template variable
	TemplateRenderer.renderAsPdf()	Object	Server-side scripts	Uses an advanced template to produce a PDF printed form
	TemplateRenderer.renderPdfToResponse()	void	Server-side scripts	Renders PDF template content as a server response
	TemplateRenderer.renderAsString()	string	Server-side scripts	Returns template content in string form
	TemplateRenderer.setTemplateById(options)	void	Server-side scripts	Sets the template using the internal ID
	TemplateRenderer.setTemplateByscriptId(options)	void	Server-side scripts	Sets the template using the script ID
Property	TemplateRenderer.templateContent	string	Server-side scripts	Content of template

N/render Module Script Sample

i Note: These sample scripts use the require function so that you can copy it into the debugger and test it. Keep in mind that you must use the define function in your entry point script (the script you attach to a script record). For additional information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

For help with writing scripts in SuiteScript 2.0, see the help topics [SuiteScript 2.0 Hello World](#) and [SuiteScript 2.0 Entry Point Script Creation and Deployment](#). For help with finding a record's internal ID, see the help topic [How do I find a record's internal ID?](#)



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to generate a PDF file from a raw XML string.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/render'],
6   function(render) {
7     function generatePdfFileFromRawXml() {
8       var xmlStr = "<?xml version='1.0'?>\n" +
9         "<!DOCTYPE pdf PUBLIC '-//big.faceless.org//report\\' \"report-1.1.dtd\\\">\n" +
10        "<pdf>\n<body font-size='18px'>\nHello World!\n</body>\n</pdf>";
11       var pdfFile = render.xmlToPdf({
12         xmlString: xmlStr
13       });
14     }
15     generatePdfFileFromRawXml();
16   });

```



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to render a transaction record into an HTML page.



Note: The entityId value in this sample is a placeholder. Before using this sample, replace the placeholder values with valid values from your NetSuite account.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/render'],
6   function(render) {
7     function renderTransactionToHtml() {
8       var transactionFile = render.transaction({
9         entityId: 23,
10        printMode: render.PrintMode.HTML
11      });
12    }
13    renderTransactionToHtml();
14  });

```



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to render an invoice into a PDF file using an XML template in the File Cabinet. This sample requires the Advanced PDF/HTML Templates feature.

```

1 /**
2  * @NApiVersion 2.x
3 */
4

```

```

5 // This sample shows how to render an invoice into a PDF file using an XML template in the file cabinet.
6 // Note that this example requires the Advanced PDF/HTML Templates feature.
7 require(['N/render', 'N/file', 'N/record'],
8     function(render, file, record) {
9         function renderRecordToPdfWithTemplate() {
10             var xmlTemplateFile = file.load('Templates/PDF Templates/invoicePDFTemplate.xml');
11             var renderer = render.create();
12             renderer.templateContent = xmlTemplateFile.getContents();
13             renderer.addRecord('grecord', record.load({
14                 type: record.Type.INVOICE,
15                 id: 37
16             }));
17             var invoicePdf = renderer.renderAsPdf();
18         }
19         renderRecordToPdfWithTemplate();
20     });

```

In the preceding sample, the invoicePDFTemplate.xml file was referenced in the File Cabinet. This file is similar to the Standard Invoice PDF/HTML Template found in Customization > Forms > Advanced PDF/HTML Templates.

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to render search results into a PDF file.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4  */
5 // This sample shows how to render search results into a PDF file.
6 define(['N/render', 'N/search'], function(render, search) {
7     function onRequest(options) {
8         var request = options.request;
9         var response = options.response;
10
11         var xmlStr = '<?xml version="1.0" encoding="UTF-8"?>\n' +
12             '<!DOCTYPE pdf PUBLIC "-//big.faceless.org//report//report-1.1.dtd">\n' +
13             '<pdf lang="ru-RU" xml:lang="ru-RU">\n' + "<head>\n" +
14             '<link name="russianfont" type="font" subtype="opentype" ' + 'src="NetSuiteFonts/verdana.ttf"' + "src-
bold="NetSuiteFonts/verdanab.ttf"' + 'src-italic="NetSuiteFonts/verdanai.ttf"' + "src-bolditalic="NetSuiteFonts/verdan
abi.ttf"' + 'bytes="2"/>\n' + "</head>\n" +
15             '<body font-family="russianfont" font-size="18">\n????????????</body>\n" + "</pdf>';
16
17         var rs = search.create({
18             type: search.Type.TRANSACTION,
19             columns: ['trandate', 'amount', 'entity'],
20             filters: []
21         }).run();
22
23         var results = rs.getRange(0, 1000);
24         var renderer = render.create();
25         renderer.templateContent = xmlStr;
26         renderer.addSearchResults({
27             templateName: 'exampleName',
28             searchResult: results
29         });
30
31         var newfile = renderer.renderAsPdf();
32         response.writeFile(newfile, false);
33     }
34
35     return {
36         onRequest: onRequest
37     };
38 });

```

render.EmailMergeResult



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates an email merge result. Use render.mergeEmail(options) to create and return this object. For a complete list of this object's properties, see EmailMergeResult Object Members .
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/render Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var mergeResult = render.mergeEmail({
4     templateId: 1234,
5     entity: {
6         type: 'employee',
7         id: 62
8     },
9     recipient: {
10        type: 'lead',
11        id: 41
12    },
13    supportCaseId: 2,
14    transactionId: 271,
15    customRecord: null
16 });
17 ...
18 //Add additional code

```

EmailMergeResult.body



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The body of the email distribution in string format
Type	string (read-only)
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/render Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 log.debug({
4   title: 'Email Body: ',
5   details: mergeResultObj.body
6 });
7 ...
8
9 //Add additional code

```

EmailMergeResult.subject



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The subject of the email distribution in string format
Type	string (read-only)
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/render Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 log.debug({
4   title: 'Email Subject: ',
5   details: mergeResultObj.subject
6 });
7 ...
8
9 //Add additional code

```

render.TemplateRenderer



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a template engine object that produces HTML and PDF printed forms utilizing advanced PDF/HTML template capabilities. The template engine object includes methods that pass in a template as string to be interpreted by FreeMarker, and render interpreted content in your choice of two different
---------------------------	--

	<p>formats: as HTML output to an <code>nlobjResponse</code> object, or as XML string that can be passed to <code>render.xmlToPdf(options)</code> to produce a PDF.</p> <p>This object is available when the Advanced PDF/HTML Templates feature is enabled.</p> <p>For a complete list of this object's methods and properties, see TemplateRenderer Object Members.</p>
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/render Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 //Advanced PDF/HTML Templates feature must be enabled
3 ...
4 var xmlTmpFile = file.load('Templates/PDF Templates/invoicePDFTemplate.xml');
5 var myFile = render.create();
6 myFile.templateContent = xmlTmpFile.getContents();
7 myFile.addRecord('record', record.load({
8   type: record.Type.INVOICE,
9   id: 37
10 }));
11 var invoicePdf = myFile.renderAsPdf();
12 ...
13 //Add additional code

```

TemplateRenderer.addCustomDataSource(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds XML or JSON as custom data source to an advanced PDF/HTML template
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.alias	string	required	Data source alias

Parameter	Type	Required / Optional	Description
options.format	render.DataSource	required	Data format
options.data	Object Document string	required	Object, document, or string

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var renderer = render.create();
4
5 var xmlObj = xml.Parser.fromString(xmlString);
6 var jsonObj = JSON.parse(jsonString);
7
8 renderer.templateContent = "${XML.book.title}<br />${XML.book.chapter[1].title}<br />${JSON.book.title}<br />${JSON.book.chapter[1].title}<br />${JSON_STR.book.title}<br />${XML_STR.book.title}";
9
10 renderer.addCustomDataSource({
11   format: render.DataSource.XML_DOC,
12   alias: "XML",
13   data: xmlObj
14 });
15 renderer.addCustomDataSource({
16   format: render.DataSource.XML_STRING,
17   alias: "XML_STR",
18   data: xmlString
19 });
20 renderer.addCustomDataSource({
21   format: render.DataSource.OBJECT,
22   alias: "JSON",
23   data: jsonObj
24 });
25 renderer.addCustomDataSource({
26   format: render.DataSource.JSON,
27   alias: "JSON_STR",
28   data: jsonString
29 });
30
31 var xml = renderer.renderAsString();
32 ...
33 //Add additional code

```

TemplateRenderer.addQuery(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Adds a SuiteAnalytics Workbook query to use as the renderer's data source. You can specify the query in two ways:</p> <ul style="list-style-type: none"> ▪ As a query.Query object (which you create using the N/query Module) using the options.query parameter. ▪ By providing the workbook ID of an existing SuiteAnalytics workbook using the options.id parameter. <p>One of options.query or options.id is required.</p> <p>This method returns a maximum of 5000 results in the query result set. If a query matches more than 5000 results, you must use options.pageIndex and options.pageSize to retrieve the full set of results.</p>
---------------------------	---

	There is no governance for this method. When the renderer is executed, rendering consumes 10 units for every iteration through the results. If the query is specified using a workbook ID, rendering consumes an additional 5 units before the first iteration through the results.
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Parent Object	render.TemplateRenderer
Sibling Object Members	TemplateRenderer Object Members
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required if options.query is not specified	Workbook query ID.
options.pageIndex	number	optional	Page index.
options.pageSize	number	optional	Page size. The minimum value is 5, and the maximum value is 1000.
options.templateName	string	required	Name of the results iterator variable referred to in the template.
options.query	query.Query object	required if options.id is not specified	Workbook query definition.

Errors

Error Code	Error Message	Thrown If
MUTUALLY_EXCLUSIVE_ARGUMENTS	Cannot use mutually exclusive arguments.	Both query and ID parameters are provided.
NEITHER_ARGUMENT_DEFINED	One of the following arguments is mandatory: id, query.	Neither options.id nor options.query are provided.
SSS_MISSING_REQD_ARGUMENT	TemplateRenderer.addQuery: Missing a required argument: options.templateName.	The template name is not specified.
UNABLE_TO_LOAD_QUERY	Unable to load query: invalidId.	The query parameter is provided but the ID is not valid.

Error Code	Error Message	Thrown If
WRONG_PARAMETER_TYPE	Wrong parameter type: options.query is expected as query.Query.	The query parameter is provided, but the ID has the incorrect type.

Syntax

Note that `TemplateRenderer.addQuery(options)` binds the results iterator to a template variable. This iterator provides sequential access to the query results. For random access to search results in FreeMarker, process your search data before rendering, and then pass the processed data to the template using `TemplateRenderer.addCustomDataSource(options)`.

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var search = query.create({
4   type: query.Type.TRANSACTION
5 });
6 var transactionlines = search.join({
7   fieldId: "transactionlines"
8 });
9 search.condition = transactionlines.createCondition({
10   fieldId: "netamount",
11   operator: query.Operator.GREATER,
12   values: 1
13 });
14 search.columns = [
15   transactionlines.createColumn({
16     fieldId: "netamount",
17     label: "Amount"
18 }),
19 ];
20
21 //get instance of TemplateRenderer - https://system.netsuite.com/app/help/helpcenter.nl?fid=section_4412065265.html
22 var renderer = render.create();
23 renderer.templateContent = "<#list page as line><item>${line['@label']}:${line[0]}</item>\n</list>";
24
25 renderer.addQuery({
26   templateName: "page",
27   query: search,
28   pageSize: 3, // minimum page size is 5, so result will contain 5 lines instead of 3
29   pageIndex: 0,
30 })
31 ...
32 //Add additional code

```

TemplateRenderer.addRecord(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Binds a record to a template variable.
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/render Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.templateName	string	required	Name of the record object variable referred to in the template
options.record	record.Record object	required	The record to add

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myContent = renderer.addRecord({
4   templateName: 'record',
5   record: record.load({
6     type: record.Type.CUSTOMER,
7     id: 1234
8   });
9 });
10 ...
11 //Add additional code

```

TemplateRenderer.addSearchResults(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Binds a search result to a template variable.
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.templateName	string	required	Name of the template

Parameter	Type	Required / Optional	Description
options.searchResult	search.Result object	required	The search result to add

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var rs = search.create({
4   type: search.Type.TRANSACTION,
5   columns: ['trandate', 'amount', 'entity'],
6   filters: []
7 }).run();
8 var results = rs.getRange(0, 1000);
9 var renderer = render.create();
10 renderer.templateContent = xmlStr;
11 renderer.addSearchResults({
12   templateName: 'exampleName',
13   searchResult: results
14 });
15 ...
16 //Add additional code

```

TemplateRenderer.renderAsPdf()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Uses the advanced template to produce a PDF printed form
Returns	file.File
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var invoicePdf = renderer.renderAsPdf();
4 ...
5 //Add additional code

```

TemplateRenderer.renderPdfToResponse()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Renders a server response into a PDF file. For example, you can pass in a response to be rendered as a PDF in a browser, or downloaded by a user.
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
response	http.ServerResponse	required	Response that will be written to PDF. For example, the response passed from a Suitelet.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var invoicePdf = renderer.renderPdfToResponse({
4   response: myServerResponseObj
5 });
6 ...
7 //Add additional code

```

TemplateRenderer.renderAsString()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Return template content in string form
Returns	string
Supported Script Types	Server-side scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var invoicePdf = renderer.renderAsString();
4 ...
5 //Add additional code

```

TemplateRenderer.renderToResponse(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Writes template content to a server response.
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.response	http.ServerResponse	required	Response to write to

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...

```

```

3 | var invoice = renderer.renderToResponse({
4 |   response: myServerResponseObj
5 | });
6 | ...
7 | //Add additional code

```

TemplateRenderer.setTemplateById(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the template using the internal ID
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	number	required	Internal ID of the template

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 | //Add additional code
2 | ...
3 | var renderer = render.create();
4 | renderer.setTemplateById(3);
5 | var xml = renderer.renderAsString();
6 | ...
7 | //Add additional code

```

For more information, see the help topics [Advanced Templates](#) and [Advanced PDF/HTML Templates](#).

To find the template ID, search for PDF Templates or Advanced PDF/HTML Templates in NetSuite.

When the list of templates is displayed, hover your cursor on the Edit or Customize link. You can also see the ID in the browser's URL when you click the link. An example of a Standard PDF template with an ID of 4 is /app/crm/common/merge/pdftemplate.nl?id=4. An example of an Advanced HTML template with an ID of 19 is /app/common/custom/advancedprint/pdftemplate.nl?id=19.

IDs from both Standard and Advanced Templates are supported.

TemplateRenderer.setTemplateByscriptId(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the template using the script ID
Returns	Void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.scriptId	string	required	Script ID of the template

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#)

```

1 //Add additional code
2 ...
3 var renderer = render.create();
4 renderer.setTemplateByscriptId({
5   scriptId: "STDMLPRICELIST"
6 });
7 var xml = renderer.renderAsString();
8 ...
9 //Add additional code

```

TemplateRenderer.templateContent

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Content of template
Type	string
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/render Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 renderer.templateContent = xmlTemplateFile.getContents();
4 ...
5 //Add additional code

```

render.bom(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Use this method to create a PDF or HTML object of a bill of material.
Returns	file.File that contains a PDF or HTML document
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/render Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.entityId	number	required	The internal ID of the bill of material to print
options.printMode	string	optional	The print output type. Set using the render.PrintMode enum. By default, uses the company/user preference for print output.
options.inCustLocale	boolean true false	optional	Applies when advanced templates are used. Print the document in the customer's locale. If basic printing is used, this parameter is ignored and the transaction form is printed in the customer's locale.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var transactionFile = render.bom({
4   entityId: 23,
5   printMode: render.PrintMode.HTML,
6   inCustLocale: true
7 });
8 ...
9 //Add additional code

```

render.create()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Use this method to produce HTML and PDF printed forms with advanced PDF/HTML templates.</p> <p>Creates render.TemplateRenderer.</p> <p>This object includes methods that pass in a template as string to be interpreted by FreeMarker, and render interpreted content in your choice of two different formats: as HTML output to http.ServerResponse, or as XML string that can be passed to render.xmlToPdf(options) to produce a PDF.</p> <p>Note: To use this method, the Advanced PDF/HTML Templates feature must be enabled.</p>
Returns	render.TemplateRenderer
Supported Script Types	<p>Server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/render Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var renderer = render.create();
4 ...
5 //Add additional code

```

render.mergeEmail(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a render.EmailMergeResult object for a mail merge with an existing scriptable email template
Returns	render.EmailMergeResult
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/render Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.templateId	number	required	Internal ID of the template
options.entity	RecordRef	required	Entity
options.recipient	RecordRef	required	Recipient
options.customRecord	RecordRef	required	Custom record
options.supportCaseId	number	required	Support case ID
options.transactionId	number	required	Transaction ID

RecordRef

You can use a RecordRef to designate the record to perform the mail merge on.

Note: The RecordRef object encapsulates the type and ID of a particular record instance.

Property	Type	Required / Optional	Description
RecordRef.id	number	required	Internal ID of the record instance
RecordRef.type	string	required	The record type ID

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myMergeResult = render.mergeEmail({}
```

```

4   templateId: 1234,
5   entity: {
6     type: 'employee',
7     id: 623
8   },
9   recipient: {
10    type: 'lead',
11    id: 543
12  },
13  supportCaseId: 674,
14  transactionId: 8987,
15  customRecord: {
16    type: 'custrecord_rewardpoints',
17    id: 5423
18  }
19 });
20 ...
21 //Add additional code

```

render.packingSlip(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Use this method to create a PDF or HTML object of a packing slip.
Returns	file.File that contains a PDF or HTML document
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/render Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.entityId	number	required	The internal ID of the packing slip to print
options.printMode	string	optional	The print output type. Set using the render.PrintMode enum. By default, uses the company/user preference for print output.
options.formId	number	optional	The packing slip form number
options.fulfillmentId	number	optional	Fulfillment ID number
options.inCustLocale	boolean true false	optional	Applies when advanced templates are used. Print the document in the customer's locale. If basic printing is used, this parameter is ignored and the transaction form is printed in the customer's locale.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var transactionFile = render.packingSlip({
4   entityId: 23,
5   printMode: render.PrintMode.HTML,
6   inCustLocale: true
7 });
8 ...
9 //Add additional code

```

render.pickingTicket(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Use this method to create a PDF or HTML object of a picking ticket.
Returns	file.File that contains a PDF or HTML document
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/render Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.entityId	number	required	The internal ID of the picking ticket to print
options.printMode	string	optional	The print output type. Set using the render.PrintMode enum. By default, uses the company/user preference for print output.
options.formId	number	optional	The packing slip form number
options.shipgroup	number	optional	Shipping group for the ticket
options.location	number	optional	Location for the ticket
options.inCustLocale	boolean true false	optional	Applies when advanced templates are used. Print the document in the customer's locale.

Parameter	Type	Required / Optional	Description
			If basic printing is used, this parameter is ignored and the transaction form is printed in the customer's locale.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var transactionFile = render.pickingTicket({
4     entityId: 23,
5     printMode: render.PrintMode.HTML,
6     inCustLocale: true
7 });
8 ...
9 //Add additional code

```

render.statement(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Use this method to create a PDF or HTML object of a statement.
Returns	file.File that contains a PDF or HTML document
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/render Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.entityId	number	required	The internal ID of the statement to print
options.printMode	string	optional	The print output type. Set using the render.PrintMode enum. By default, uses the company/user preference for print output.
options.formId	number	optional	Internal ID of the form to use to print the statement

Parameter	Type	Required / Optional	Description
options.startDate	Date	optional	Date of the oldest transaction to appear on the statement
options.statementDate	Date	optional	Statement date
options.openTransactionsOnly	boolean true false	optional	Include only open transactions
options.inCustLocale	boolean true false	optional	Applies when advanced templates are used. Print the document in the customer's locale. If basic printing is used, this parameter is ignored and the transaction form is printed in the customer's locale.
options.consolidateStatements	boolean true false	optional	Convert all amount values to the base currency

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var transactionFile = render.statement({
4   entityId: 23,
5   printMode: render.PrintMode.HTML,
6   inCustLocale: true
7 });
8 ...
9 //Add additional code

```

render.transaction(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Use this method to create a PDF or HTML object of a transaction.
	<p>Note: File size is limited to 10MB.</p> <p>If the Advanced PDF/HTML Templates feature is enabled, you can associate an advanced template with the custom form saved for a transaction. The advanced template is used to format the printed transaction. For details about this feature, see the help topic Advanced PDF/HTML Templates</p>
Returns	<code>file.File</code> that contains a PDF or HTML document
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/render Module

Since	2015.2
--------------	--------

Parameters

Parameter	Type	Required / Optional	Description
options.entityId	number	required	The internal ID of the transaction to print
options.printMode	enum	optional	The print output type. Set using the render.PrintMode enum. By default, uses the company/user preference for print output.
options.formId	number	optional	The transaction form number
options.inCustLocale	boolean true false	optional	Applies when advanced templates are used. Print the document in the customer's locale. If basic printing is used, this parameter is ignored and the transaction form is printed in the customer's locale.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var transactionFile = render.transaction({
4   entityId: 23,
5   printMode: render.PrintMode.HTML,
6   inCustLocale: true
7 });
8 ...
9 //Add additional code

```

render.xmlToPdf(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to pass XML to the Big Faceless Organization (BFO) tag library (which is stored by NetSuite), and return a PDF file. BFO is used in NetSuite. For version details, see the help topic Third-Party Products Used in Advanced Printing .
Note: File size cannot exceed 10MB.	
Returns	file.File
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	10 units
Module	N/render Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.xmlString	xml.Document string	required	XML document or string to convert to PDF. For information and examples about using BFO tags to create this document or string, see the help topic nlapiXMLToPDF(xmlstring) , which documents the corresponding SuiteScript 1.0 API.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var pdfFile = render.xmlToPdf({
4   xmlString: xmlStr
5 });
6 ...
7 //Add additional code

```

render.DataSource

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

 **Note:** JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported data source types. Use this enum to set the options.format parameter.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/render Module

Values

- JSON

- OBJECT
- XML_DOC
- XML_STRING

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 renderer.addCustomDataSource({
4   format: render.DataSource.JSON,
5   alias: "JSON_STR",
6   data: jsonString
7 });
8 ...
9 //Add additional code

```

render.PrintMode



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported print output types. Use this enum to set the options.printMode parameter.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/render Module

Values

- DEFAULT
- HTML
- PDF

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/render Module Script Sample](#).

```

1 //Add additional code
2 ...
3 printMode: render.PrintMode.HTML
4 ...
5 //Add additional code

```

N/runtime Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the runtime module when you want to view runtime settings for the script, the session, or the user. You can also use this module to set a session key and to see whether a particular feature is enabled in your account.

- [N/runtime Module Members](#)
- [Script Object Members](#)
- [Session Object Members](#)
- [User Object Members](#)
- [N/runtime Module Script Samples](#)

N/runtime Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	runtime.Script	Object	Client and server scripts	Encapsulates the runtime settings of the currently executing script.
	runtime.Session	Object	Client and server scripts	Encapsulates the user session for the currently executing script.
	runtime.User	Object	Client and server scripts	Encapsulates the properties and preferences of the user currently executing the script.
Method	runtime.getCurrentScript()	runtime.Script	Client and server scripts	Returns a runtime.Script object that represents the currently executing script.
	runtime.getCurrentSession()	runtime.Session	Client and server scripts	Returns a runtime.Session object that represents the user session for the currently executing script.
	runtime.getCurrentUser()	runtime.User	Client and server scripts	Returns a runtime.User object that represents the properties and preferences of the user currently executing the script.
	runtime.isFeatureInEffect(options)	boolean	Client and server scripts	Indicates whether a particular feature is enabled in a NetSuite account. These are the features that appear on the Enable Features page.
Property	runtime.accountId	string (read-only)	Client and server scripts	The account ID for the current user.
	runtime.envType	string (read-only)	Client and server scripts	The current environment in which the script is executing. This property uses values from the runtime.EnvType enum.
	runtime.executionContext	string (read-only)	Client and server scripts	The trigger of the current script. This property uses values from the runtime.ContextType enum.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	runtime.processorCount	number (read-only)	Client and server scripts	The number of processors available to the current account.
	runtime.queueCount	number (read-only)	Client and server scripts	The number of scheduled script queues available to the current account.
	runtime.version	string (read-only)	Client and server scripts	The version of NetSuite that the method is called in. For example, this property in an account running NetSuite 2015.2 is 2015.2.
Enum	runtime.ContextType	enum	Client and server scripts	Holds the context values for script triggers. This is the type for the runtime.executionContext property.
	runtime.EnvType	enum	Client and server scripts	Holds all possible environment types that the current script can execute in. This is the type for the runtime.envType property.
	runtime.Permission	enum	Client and server scripts	Holds the user permission level for a specific permission ID. This is the type returned by the User.getPermission(options) method.

Script Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Script.getParameter(options)	number Date string boolean	Client and server scripts	Returns the value of a script parameter for the currently executing script.
	Script.getRemainingUsage()	number	Client and server scripts	Returns the number of units remaining (per governance limitations) for the currently executing script.
Property	Script.apiVersion	string (read-only)	Client and server scripts	The current script's runtime version.
	Script.bundleIds	Array (read-only)	Client and server scripts	An array of bundle IDs for the bundles that include the currently executing script.
	Script.deploymentId	string (read-only)	Server scripts	The deployment ID for the script deployment of the currently executing script.
	Script.id	string (read-only)	Client and server scripts	The script ID for the currently executing script.
	Script.logLevel	string (read-only)	Server scripts	The script logging level for the currently executing script.
	Script.percentComplete	number	Client and server scripts	The percent complete for the current scheduled script execution. This value will appear

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				in the % Complete column on the Scheduled Script Status page.

Session Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Session.get(options)	string null	Server scripts	Returns the user-defined session object value associated with the session object key. Both the session object value and associated key are defined using Session.set(options) .
	Session.set(options)	void	Server scripts	Sets a key and value for a user-defined session object.

User Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	User.getPermission(options)	string	Client and server scripts	Returns a runtime.Permission user permission level for the specified permission.
	User.getPreference(options)	string	Client and server scripts	Returns the value of a NetSuite preference. Currently only General Preferences and Accounting Preferences are exposed in SuiteScript. For more information about these preferences, see the help topics General Preferences and Accounting Preferences .
Property	User.contact	number(read-only)	Client and server scripts	The internal ID of the currently logged-in contact.
	User.department	number (read-only)	Client and server scripts	The internal ID of the department for the current user.
	User.email	string (read-only)	Client and server scripts	The email address of the current user.
	User.id	number (read-only)	Client and server scripts	The internal ID of the current user.
	User.location	number (read-only)	Client and server scripts	The internal ID of the location of the current user.
	User.name	string (read-only)	Client and server scripts	The name of the current user.
	User.role	number (read-only)	Client and server scripts	The internal ID of the role for the current user.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	User.roleCenter	string (read-only)	Client and server scripts	The string value of the center type, or role center, for the current user.
	User.roleId	string (read-only)	Client and server scripts	The custom scriptId of the role for the current user.
	User.subsidiary	number (read-only)	Client and server scripts	The internal ID of the subsidiary for the current user.

N/runtime Module Script Samples

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to use a Suitelet to write user and session information for the currently executing script to the response.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4 */
5 //Write user and session information for the currently executing script to the response.
6 define(['N/runtime'], function(runtime) {
7     function onRequest(context) {
8         var remainingUsage = runtime.getCurrentScript().getRemainingUsage();
9         var userRole = runtime.getCurrentUser().role;
10        runtime.getCurrentSession().set({
11            name: 'scope',
12            value: 'global'
13        });
14        var sessionScope = runtime.getCurrentSession().get({
15            name: 'scope'
16        });
17        log.debug('Remaining Usage:', remainingUsage);
18        log.debug('Role:', userRole);
19        log.debug('Session Scope:', sessionScope);
20        context.response.write('Executing under role: ' + userRole
21            + '. Session scope: ' + sessionScope + '.');
22    }
23
24    return {
25        onRequest: onRequest
26    };
27});

```

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to use a scheduled script to create multiple sales records and logs the record creation progress.

```

1 /**
2  * @NApiVersion 2.x

```

```

3 * @NScriptType ScheduledScript
4 */
5
6 //Create multiple sales records and log the record creation progress.
7 define(['N/runtime', 'N/record'], function(runtime, record){
8     return {
9         execute: function(context) {
10             var script = runtime.getCurrentScript();
11             for (x = 0; x < 500; x++) {
12                 var rec = record.create({
13                     type: record.Type.SALES_ORDER
14                 });
15                 script.percentComplete = (x * 100)/500;
16                 log.debug({
17                     title: 'New Sales Orders',
18                     details: "Record creation progress: " + script.percentComplete + "%"
19                 });
20             }
21         }
22     });
23 });

```

runtim.Script

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the runtime settings of the currently executing script. Use runtime.getCurrentScript() to access this object. For a complete list of this object's methods and properties, see Script Object Members .
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript(); //scriptObj is a runtim.Script object
4 log.debug('Script ID: ' + scriptObj.id);
5 ...
6 // Add additional code

```

Script.getParameter(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of a script parameter for the currently executing script.
---------------------------	---

	For information on script parameters, see the help topic Creating Script Parameters Overview .
Returns	number Date string boolean
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/runtime Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	The name of the script parameter.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	The options.name parameter is not specified.
WRONG_PARAMETER_TYPE	The value for the options.name parameter is not a string.

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/runtime Module Script Samples .
---	---

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript();
4 log.debug('Script parameter of custscript1: ' +
5   scriptObj.getParameter({name: 'custscript1'}));
6 ...
7 // Add additional code

```

Script.getRemainingUsage()

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Method Description	Returns the number of units remaining (per governance limitations) for the currently executing script. For more information, see the help topic SuiteScript Governance .
Returns	number

Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/runtime Module
Since	2015.2



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript();
4 log.debug('Remaining governance units: ' + scriptObj.getRemainingUsage());
5 ...
6 // Add additional code

```

Script.apiVersion



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The current script's runtime version.
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript();
4 var scriptApiVersion = scriptObj.apiVersion;
5 ...
6 // Add additional code

```

Script.bundleIds



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of bundle IDs for the bundles that include the currently executing script.
-----------------------------	---

	<p>When you use this property, consider the following:</p> <ul style="list-style-type: none"> ■ The array can contain any number of bundle IDs, because a bundle installation script can be associated with multiple bundles. ■ The array does not contain bundle IDs of deprecated bundles. ■ The elements in the array are sorted in ascending order from the lowest bundle ID to the highest bundle ID that includes the currently executing script. <p>However, it is not guaranteed that the last element in the array corresponds to the bundle that triggered the currently executing script during the bundle installation.</p>
Type	string [] (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript();
4 var bundleArr = scriptObj.bundleIds;
5 ...
6 // Add additional code

```

Script.deploymentId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The deployment ID for the script deployment on the currently executing script.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript();

```

```

4 log.debug('Deployment Id: ' + scriptObj.deploymentId);
5 ...
6 // Add additional code

```

Script.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The script ID for the currently executing script.
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript();
4 log.debug('Script Id: ' + scriptObj.id);
5 ...
6 // Add additional code

```

Script.logLevel



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The script logging level for the currently executing script. This method is not supported on client scripts. Returns one of the following values: <ul style="list-style-type: none">■ DEBUG■ AUDIT■ ERROR■ EMERGENCY
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript();
4 log.debug('Logging level: ' + scriptObj.logLevel);
5 ...
6 // Add additional code

```

Script.percentComplete



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The percent complete specified for the current scheduled script execution. This value appears in the % Complete column on the Scheduled Script Status page. This value can be set or retrieved.
Type	number
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_OPERATION_UNAVAILABLE		The currently executing script is not a scheduled script.

Syntax



Important: The following code snippets show the syntax for this member. They are not a functional examples. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // Gets the percentage of records completed
4 var scriptObj = runtime.getCurrentScript();
5 if (scriptObj.executionContext === ContextType.SCHEDULED) {
6   log.debug({
7     details: 'Script percent complete: ' + scriptObj.percentComplete
8   });
9 }
10 ...
11 // Add additional code

```

```

1 // Add additional code
2 ...
3 // Sets the percent complete
4 var script = runtime.getCurrentScript();

```

```

5  for (x = 0; x < 500; x++) {
6    var rec = record.create({
7      type: record.Type.SALES_ORDER
8    });
9    script.percentComplete = (x * 100)/500;
10   log.debug({
11     title: 'New Sales Orders',
12     details: 'Record creation progress: ' + script.percentComplete + '%'
13   });
14 }
15 ...
16 // Add additional code

```

runtim.Session



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>Encapsulates the user session for the currently executing script.</p> <p>Use this object to set and get user-defined objects for the current user session. Use the objects to track user-related session data. For example, you can gather information about the user scope, budget, or business problems.</p> <p>Use Session.set(options) to set session object values. Use Session.get(options) to retrieve session object values.</p> <p>For a complete list of this object's methods, see Session Object Members.</p>
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var sessionObj = runtime.getCurrentSession(); //sessionObj is a runtim.Session object
4 sessionObj.set({
5   name: 'myKey',
6   value: 'myValue'
7 });
8 log.debug('Session object myKey value: ' + sessionObj.get({name: 'myKey'}));
9 ...
10 // Add additional code

```

Session.get(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Returns the user-defined session object value associated with a session object key.</p> <p>Both the session object value and associated key are defined using Session.set(options).</p>
---------------------------	--

	If the key does not exist, this method returns null.
Returns	string null
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/runtime Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	Key used to store the session object.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	The options.name parameter is not specified.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var sessionObj = runtime.getCurrentSession();
4 sessionObj.set({
5   name: 'myKey',
6   value: 'myValue'
7 });
8 log.debug('Session object myKey value: ' + sessionObj.get({name: 'myKey'}));
9 ...
10 // Add additional code

```

Session.set(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets a key and value for a user-defined session object. Use Session.get(options) to retrieve the object value after you set it.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/runtime Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	Key used to store the session object.	2015.2
options.value	string	required	Value to associate with the key in the user session.	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	The options.name or options.value parameter is not specified.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var sessionObj = runtime.getCurrentSession();
4 sessionObj.set({
5   name: 'myKey',
6   value: 'myValue'
7 });
8 log.debug('Session object myKey value: ' + sessionObj.get({name: 'myKey'}));
9 ...
10 // Add additional code

```

runtim.User

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the properties and preferences of the user currently executing the script. For a complete list of this object's methods and properties, see User Object Members .
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 ...
5 // Add additional code

```

User.getPermission(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a runtime.Permission user permission level for the specified permission.
Returns	string
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/runtime Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	required	Internal ID of a permission. For a list of permission IDs, see Permission Names and IDs .	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 var userPermission = userObj.getPermission({
5   name: 'ADMI_ACCOUNTING'
6 });
7 log.debug('User permission of ADMI_ACCOUNTING: ' +
8           (userPermission ===
9            runtime.Permission.FULL ? 'FULL' : userPermission));
10 ...
11 // Add additional code

```

User.getPreference(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Returns the value set for a NetSuite preference.</p> <p>Currently only General Preferences and Accounting Preferences are exposed in SuiteScript.</p> <p>You can also view General Preferences by going to Setup > Company > General Preferences. View Accounting Preferences by going to Setup > Accounting > Accounting Preferences.</p> <p>If you want to change the value of a General or Accounting preference using SuiteScript 2.0, you must load each preference page using <code>config.load(options)</code>, where <code>options.name</code> is <code>COMPANY_PREFERENCES</code> or <code>ACCOUNTING_PREFERENCES</code>. The <code>config.load(options)</code> method returns a <code>Record</code>. You can use the <code>Record.setValue(options)</code> method to set the preference.</p> <p>For more information about these preferences, see the help topics General Preferences and Accounting Preferences.</p>
	<p>Note: The permission level will be <code>Permission.FULL</code> if the script is configured to execute as admin. You can configure a script to execute as admin by selecting "administrator" from the Execute as Role field on Script Deployment page.</p>
Returns	<code>string</code>
Supported Script Types	<p>Client and server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/runtime Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.name</code>	<code>string</code>	required	Internal ID of the preference. For a list of preference IDs, see Permission Names and IDs .	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 var userPref = userObj.getPreference ({
5   name: 'emailemployeeonapproval'
6 });
7 log.debug('User preference for emailemployeeonapproval: ' + userPref);
8 ...
9 // Add additional code

```

User.contact



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of the currently logged-in contact. If no logged-in entity or other entity than contact is logged in, then 0 is returned as value.
Type	number (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2019.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Internal ID of current contact: ' + userObj.contact);
5 ...
6 // Add additional code

```

User.department



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of the department for the current user.
Type	number (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Internal ID of current user department: ' + userObj.department);
5 ...
6 // Add additional code

```

User.email



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The email address of the current user. To use this property, the email field on the user employee record must contain an email address.
	Note: In a shopping context where the shopper is recognized but not logged in, this method can be used to return the shopper's email, instead of getting it from the customer record.
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Current user email: ' + userObj.email);
5 ...
6 // Add additional code

```

User.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of the current user.
Type	number (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Internal ID of current user: ' + userObj.id);
5 ...
6 // Add additional code

```

User.location

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of the location of the current user.
Type	number (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Internal ID of current user location: ' + userObj.location);
5 ...
6 // Add additional code

```

User.name

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of the current user.
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Name of current user: ' + userObj.name);
5 ...
6 // Add additional code

```

User.role



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of the role for the current user.
Type	number (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Internal ID of current user role: ' + userObj.role);
5 ...
6 // Add additional code

```

User.roleCenter



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The string value of the center type, or role center, for the current user. The NetSuite UI adjusts automatically to different users' business needs. For each user, NetSuite displays a variable set of tabbed pages, called a center , based on the user's assigned role. Each NetSuite center provides, for users with related roles, the pages and links they need to do their jobs. For more information about NetSuite centers, see the help topic Centers Overview .
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('String value of current user center type (role center): ' + userObj.roleCenter);
5 ...
6 // Add additional code

```

User.roleId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The custom scriptId of the role for the current user. You can use this value instead of User.role . When bundling a custom role, the internal ID number of the role in the target account can change after the bundle is installed. Therefore, in the target account you can use this property to access the unique/custom scriptId assigned to the role.
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Custom script ID of current user role: ' + userObj.roleId);
5 ...
6 // Add additional code

```

User.subsidiary



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of the subsidiary for the current user.
Type	number (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/runtime Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();
4 log.debug('Internal ID of current user subsidiary: ' + userObj.subsidiary);
5 ...
6 // Add additional code

```

runtime.getCurrentScript()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a runtime.Script object that represents the currently executing script. Use this method to get properties and parameters of the currently executing script and script deployment. If you want to get properties for the session or user, use runtime.getCurrentSession() or runtime.getCurrentUser() instead.
Returns	runtime.Script
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/runtime Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var scriptObj = runtime.getCurrentScript();
4 ...
5 // Add additional code

```

runtime.getCurrentSession()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a runtime.Session object that represents the user session for the currently executing script.
---------------------------	---

	Use this method to get session objects for the current user session. If you want to get properties for the script or user, use runtime.getCurrentScript() or runtime.getCurrentUser() instead.
Returns	runtime.Session
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/runtime Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var sessionObj = runtime.getCurrentSession();
4 ...
5 // Add additional code

```

runtime.getCurrentUser()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a runtime.User object that represents the properties and preferences for the user currently executing the script. Use this method to get session objects for the current user session. If you want to get properties for the script or session, use runtime.getCurrentScript() or runtime.getCurrentUser() instead.
Returns	runtime.User
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/runtime Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var userObj = runtime.getCurrentUser();

```

```

4 ...
5 // Add additional code

```

runtime.isFeatureInEffect(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Use this method to determine if a particular feature is enabled in a NetSuite account. These are the features that appear on the Enable Features page at Setup > Company > Enable Features.
Returns	boolean
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/runtime Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.feature	string	required	The internal ID of the feature to check. For a list of feature internal IDs, see the help topic Feature Names and IDs .	2015.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	The options.feature parameter is not specified.
WRONG_PARAMETER_TYPE	The value for the options.feature parameter is not a string.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var featureInEffect = runtime.isFeatureInEffect({
4   feature: 'ADVBILLING'
5 });
6 log.debug('Advanced Billing feature is enabled: ' + featureInEffect);
7 ...
8 // Add additional code

```

runtime.accountId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The account ID for the current user.
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug('Account ID for the current user: ' + runtime.accountId);
4 ...
5 // Add additional code

```

runtime.envType



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The current environment in which the script is executing. This property uses values from the runtime.EnvType enum.
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug('Environment for current user: ' + JSON.stringify(runtime.envType));
4 ...

```

```
5 | // Add additional code
```

runtime.executionContext



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The execution context trigger of the current script. Execution contexts provide information about how a script is triggered to execute. For example, a script can be triggered in response to an action in the NetSuite application, or an action occurring in another context, such as a web services integration. You can use execution context filtering to ensure that your scripts are triggered only when necessary. For more information, see the help topic Execution Contexts.</p> <p>You can use the <code>runtime.executionContext</code> property to determine the execution context for a script and choose different logic depending on the context. Consider a script that applies to customer records and uses the <code>beforeLoad(scriptContext)</code> entry point. You may not want to execute this script when the entry point is triggered in response to a REST web services request. To prevent the script from executing in this context, you can do the following:</p> <pre>1 if (runtime.executionContext === runtime.ContextType.REST_WEBSERVICES) { 2 return; 3 }</pre> <p>This property uses values from the <code>runtime.ContextType</code> enum.</p>
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```
1 | // Add additional code
2 |
3 | if (runtime.executionContext === runtime.ContextType.USEREVENT)
4 |   return;
5 |
6 | // Add additional code
```

runtime.processorCount



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The number of processors available to the current account.
-----------------------------	--

	<p>SuiteCloud Processors is the current system used to execute (process) scheduled scripts and map/reduce scripts. This property is helpful if you are a SuiteApp developer and your script needs to know the total number of processors available to a deployment.</p> <p>For scheduled script deployments that continue to use queues, use runtime.queueCount. With the introduction of SuiteCloud Processors, map/reduce script deployments and new scheduled script deployments no longer use queues, but pre-existing scheduled script deployments continue to use queues until the queues are removed (see the help topic SuiteCloud Processors – Supported Task Types).</p> <p>Be aware that the number of processors available may not be the same as the number of queues available. For more information, see the help topic SuiteCloud Plus Settings.</p> <p>Note: The <code>runtime.processorCount</code> property reflects the number of processors available to an account. It is not impacted by changes to deployments. The value is the same regardless of whether deployments continue to use queues. For more information, see the help topic SuiteCloud Processors – Supported Task Types.</p>
Type	number (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2018.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug('Number of processors available: ' + runtime.processorCount);
4 ...
5 // Add additional code

```

runtime.queueCount



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The number of scheduled script queues available to the current account.</p> <p>SuiteCloud Processors is the current system used to execute (process) scheduled scripts and map/reduce scripts. This property is helpful if you are a SuiteApp developer and your script needs to know the total number of queues available to a deployment.</p> <p>For map/reduce script deployments, use runtime.processorCount. With the introduction of SuiteCloud Processors, no map/reduce script deployments use queues (see the help topic SuiteCloud Processors – Supported Task Types).</p> <p>Be aware that the number of queues available may not be the same as the number of processors available (see the help topic SuiteCloud Plus Settings).</p>
-----------------------------	--

	<p>Note: If all scheduled script deployments in an account are configured to no longer use queues (see the help topic SuiteCloud Processors – Supported Task Types), the value of <code>runtime.queueCount</code> is unchanged. This property reflects the number of queues available to an account. It is not impacted by changes to deployments.</p>
	For more information on scheduled scripts, see the help topic SuiteScript 2.0 Scheduled Script Type .
Type	number (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug('Number of queues available: ' + runtime.queueCount);
4 ...
5 // Add additional code

```

runtime.version



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The version of NetSuite that the method is called in. For example, the <code>runtime.version</code> property in an account running NetSuite 2015.2 is 2015.2. For example, you can use this method when installing a bundle in another NetSuite accounts to find out the version number before installing the bundle.
Type	string (read-only)
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/runtime Module Script Samples](#).

```

1 // Add additional code
2 ...
3 log.debug('Current NetSuite version: ' + runtime.version);

```

```

4 | ...
5 // Add additional code

```

runtime.ContextType



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the execution context values for script triggers. This is the type for the runtime.executionContext property.
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Values

For a description of each execution context value, see the help topic [Execution Context Types](#).

Value	Sets <code>runtime.executionContext</code> Property To
ACTION	ACTION
ADVANCEDREVREC	ADVANCEDREVREC
BANKCONNECTIVITY	BANKCONNECTIVITY
BANKSTATEMENTPARSER	BANKSTATEMENTPARSER
BUNDLE_INSTALLATION	BUNDLEINSTALLATION
CLIENT	CLIENT
CONSOLRATEADJUSTOR	CONSOLRATEADJUSTOR
CSV_IMPORT	CSVIMPORT
CUSTOMGLLINES	CUSTOMGLLINES
CUSTOM_MASSUPDATE	CUSTOMMASSUPDATE
DEBUGGER	DEBUGGER
EMAIL_CAPTURE	EMAILCAPTURE
FICONNECTIVITY	FICONNECTIVITY
MAP_REDUCE	MAPREDUCE
NONE	NONE
PAYMENTGATEWAY	PAYMENTGATEWAY

Value	Sets runtime.executionContext Property To
PAYMENTPOSTBACK	PAYMENTPOSTBACK
PLATFORMEXTENSION	PLATFORMEXTENSION
PORTLET	PORTLET
PROMOTIONS	PROMOTIONS
RESTLET	RESTLET
REST_WEBSERVICES	RESTWEBSERVICES
SCHEDULED	SCHEDULED
SDF_INSTALLATION	SDFINSTALLATION
SHIPPING_PARTNERS	SHIPPINGPARTNERS
SUITELET	SUITELET
TAX_CALCULATION	TAXCALCULATION
USEREVENT	USEREVENT
USER_INTERFACE	USERINTERFACE
WEBAPPLICATION	WEBAPPLICATION
WEBSERVICES	WEBSERVICES
WEBSTORE	WEBSTORE
WORKFLOW	WORKFLOW

runtime.EnvType

Note: The content in this help topic pertains to SuiteScript 2.0.

Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds all possible environment types that the current script can execute in. This is the type for the runtime.envType property.
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Values

- SANDBOX

- PRODUCTION
- BETA
- INTERNAL

runtime.Permission

- Note:** The content in this help topic pertains to SuiteScript 2.0.
- Note:** JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the user permission level for a specific permission ID. This is the type returned by the User.getPermission(options) method. For information on working with NetSuite permissions, see the help topics NetSuite Permissions Overview and Permission Names and IDs .
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/runtime Module
Since	2015.2

Values

- FULL
- EDIT
- CREATE
- VIEW
- NONE

N/search Module

- Note:** The content in this help topic pertains to SuiteScript 2.0.

Load the search module to create and run on-demand or saved searches and analyze and iterate through the search results. You can use this module to do the following:

- Search for a single record using keywords
- Create and save searches
- Load and run previously saved searches
- Search for duplicate records
- Return a set of records that match filter criteria you define

You can also paginate search results and construct navigation that jumps between the next and previous pages. Due to the performance benefits, this is a suitable approach for working with a large result set.

In this help topic

- [N/search Module Members](#)

- [Search Object Members](#)
- [Result Object Members](#)
- [Column Object Members](#)
- [Filter Object Members](#)
- [Page Object Members](#)
- [PagedData Object Members](#)
- [PageRange Object Members](#)
- [ResultSet Object Members](#)
- [N/search Module Script Samples](#)

N/search Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	search.Search	Object	Client and server-side scripts	Encapsulates a NetSuite search. Use the methods available to the Search object to create a search, run a search, or save a search.
	search.Result	Object	Client and server-side scripts	Encapsulate a single search result row. Use the methods and properties for the Result object to get the column values for the result row.
	search.Column	Object	Client and server-side scripts	Encapsulates a single search column in a search.Search object. Use the methods and properties available to the Column object to get or set Column properties.
	search.Filter	Object	Client and server-side scripts	Encapsulates a search filter used in a search. Use the properties for the Filter object to get and set the filter properties.
	search.ResultSet	Object	Client and server-side scripts	Encapsulates a set of search results returned by Search.run() .
	search.Page	Object	Client and server-side scripts	Encapsulates a set of search results for a single search page.
	search.PagedData	Object	Client and server-side scripts	Holds metadata about a paginated query.
	search.PageRange	Object	Client and server-side scripts	Defines the page range to bound the result set for a paginated query.
	search.Settings	Object	Client and server-side scripts	Encapsulates a search setting. Search settings let you specify search parameters that are typically available only in the UI.
Method	search.create(options)	search.Search	Client and server-side scripts	Creates a new search and returns it as a search.Search object.
	search.create.promise(options)	search.Search	Client scripts	Creates a new search asynchronously and returns it as a search.Search object.
	search.load(options)	search.Search	Client and server-side scripts	Loads an existing saved search and returns it as a search.Search object.
	search.load.promise(options)	search.Search	Client scripts	Loads an existing saved search asynchronously and returns it as a search.Search object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	search.delete(options)	void	Client and server-side scripts	Deletes an existing saved search asynchronously and returns it as a search.Search object.
	search.delete.promise(options)	void	Client scripts	Deletes an existing saved search and returns it as a search.Search object.
	search.duplicates(options)	search.Result []	Client and server-side scripts	Performs a search for duplicate records based on the duplicate detection configuration for the account. Returns an array of search.Result objects.
	search.duplicates.promise(options)	search.Result []	Client scripts	Performs a search for duplicate records asynchronously based on the duplicate detection configuration for the account. Returns an array of search.Result objects.
	search.global(options)	search.Result []	Client and server-side scripts	Performs a global search against a single keyword or multiple keywords.
	search.global.promise(options)	search.Result []	Client scripts	Performs a global search asynchronously against a single keyword or multiple keywords.
	search.lookupFields(options)	Object array	Client and server-side scripts	Performs a search for one or more body fields on a record. Returns select fields as an object with value and text properties. Returns multiselect fields as an object with value:text pairs.
	search.lookupFields.promise(options)	Object array	Client scripts	Performs a search asynchronously for one or more body fields on a record. Returns select fields as an object with value and text properties. Returns multiselect fields as an object with value:text pairs.
	search.createColumn(options)	search.Column	Client and server-side scripts	Creates a new search column as a search.Column object.
	search.createFilter(options)	search.Filter	Client and server-side scripts	Creates a new search filter as a search.Filter object.
	search.createSetting(options)	search.Setting	Client and server-side scripts	Creates a new search setting and returns it as a search.Setting object.
Enum	search.Operator	enum	Client and server-side scripts	Enumeration that holds the values for search operators to use with the search.Filter object.
	search.Sort	enum	Client and server-side scripts	Enumeration that holds the values for supported sorting directions used with search.createColumn(options) .
	search.Summary	enum	Client and server-side scripts	Enumeration that holds the values for summary types used by the Column.summary object.
	search.Type	enum	Client and server-side scripts	Enumeration that holds the string values for search types supported in the N/search Module . This enum is used to pass the type argument to search.create(options) .

Search Object Members

The following members are called on [search.Search](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Search.save()	number	Client and server-side scripts	Saves a search created by search.create(options) or loaded with search.load(options) . Returns the internal ID of the saved search.
	Search.save.promise()	number	Client scripts	Asynchronously saves a search created by search.create(options) or loaded with search.load(options) . Returns the internal ID of the saved search.
	Search.run()	search.ResultSet	Client and server-side scripts	Runs an on demand search created with search.create(options) or a search loaded with search.load(options) , returning the results as a search.ResultSet .
	Search.runPaged(options)	search.PagedData	Client and server-side scripts	Runs the current search and returns a search.PagedData Object.
	Search.runPaged.promise(options)	search.PagedData	Client scripts	Asynchronously runs the current search and returns a search.PagedData Object.
Property	Search.searchType	string	Client and server-side scripts	Search type on which a search is based.
	Search.searchId	number	Client and server-side scripts	Internal ID of a search.
	Search.filters	search.Filter[]	Client and server-side scripts	Filters for the search as an array of search.Filter objects.
	Search.filterExpression	Object[]	Client and server-side scripts	Search filter expression for the search as an array of expression objects.
	Search.columns	search.Column[] string[]	Client and server-side scripts	Columns to return for this search as an array of search.Column objects or a string array of column names.
	Search.packageId	string	Client and server-side scripts	The application ID for the search.
	Search.settings	search.Setting[] string[]	Client and server-side scripts	Search settings for this search as an array of search.Setting objects or a string array of column names.
	Search.title	string	Client and server-side scripts	Title for a saved search. Use this property to set the title for a search before you save it for the first time.
	Search.id	string	Client and server-side scripts	Script ID for a saved search, starting with customsearch.
	Search.isPublic	boolean true false	Client and server-side scripts	Value is true if the search is public, or false if it is not.

Column Object Members

The following members are called on [search.Column](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Column.setWhenOrderedBy(options)	search.Column	Client and server-side scripts	Returns the search column for which the minimal or maximal

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				value should be found when returning the search.Column value.
Property	Column.name	string (read-only)	Client and server-side scripts	Name of a search column as a string.
	Column.join	string (read-only)	Client and server-side scripts	Join ID for a search column as a string.
	Column.summary	string (read-only)	Client and server-side scripts	Returns the summary type for a search column.
	Column.formula	string	Client and server-side scripts	Formula used for a search column as a string.
	Column.label	string	Client and server-side scripts	Label used for the search column. You can only get or set custom labels with this property.
	Column.function	string	Client and server-side scripts	Special function used in the search column as a string.

Filter Object Members

The following members are called on [search.Filter](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	Filter.name	string (read-only)	Client and server-side scripts	Name or internal ID of the search field.
	Filter.join	string (read-only)	Client and server-side scripts	Join ID for the search filter.
	Filter.operator	string (read-only)	Client and server-side scripts	Operator used for the search filter.
	Filter.summary	search.Summary	Client and server-side scripts	Summary type for the search filter.
	Filter.formula	string	Client and server-side scripts	Formula used by the search filter.

Page Object Members

The following members are called on the [search.Page](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Page.next()	void	Client and server-side scripts	Gets the next segment of data from a paginated search

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Page.next.promise()	void	Client scripts	Asynchronously gets the next segment of data from a paginated search
	Page.prev()	void	Client and server-side scripts	Gets the previous segment of data from a paginated search
	Page.prev.promise()	void	Client scripts	Asynchronously gets the previous segment of data from a paginated search
Property	Page.data	search.Result[]	Client and server-side scripts	The results from a paginated search.
	Page.isFirst	read-only boolean	Client and server-side scripts	Indicates whether a page is the first page of data for a result set
	Page.isLast	read-only boolean	Client and server-side scripts	Indicates whether a page is the last page of data for a result set.
	PagepagedData	read-only search.PagedData	Client and server-side scripts	The PagedData Object used to fetch this Page Object.
	Page.pageRange	read-only search.PageRange	Client and server-side scripts	The PageRange Object used to fetch this Page Object.

PagedData Object Members

The following members are called on [search.PagedData](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	PagedData.fetch(options)	search.Page	Client and server-side scripts	Retrieves the data within the specified page range.
	PagedData.fetch.promise()	search.Page	Client scripts	Asynchronously retrieves the data within the specified page range.
Property	PagedData.count	read-only number	Client and server-side scripts	The total number of results when Search.runPaged(options) was executed.
	PagedData.pageRanges	read-only search.PageRange[]	Client and server-side scripts	The collection of PageRange objects that divide the entire result set into smaller groups.
	PagedData.pageSize	read-only number	Client and server-side scripts	The maximum number of entries per page
	PagedData.searchDefinition	read-only search.Search	Client and server-side scripts	The search criteria used when Search.runPaged(options) was executed.

PageRange Object Members

The following members are called on [search.PageRange](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	PageRange.compoundLabel	read-only string	Client and server-side scripts	Human-readable label with beginning and ending range identifiers
	PageRange.index	read-only number	Client and server-side scripts	The index of this page range.

Result Object Members

The following members are called on [search.Result](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Result.getValue(options)	string	Client and server-side scripts	Used on formula fields and non-formula (standard) fields to get the value of a specified search return column.
	Result.getValue(column)	string	Client and server-side scripts	Used on formula and non-formula (standard) fields. Returns the string value of a specified search result column. For convenience, this method takes a single search.Column Object.
	Result.getText(column)	string	Client and server-side scripts	The text value for a search.Column if it is a stored select field.
	Result.getText(options)	string	Client and server-side scripts	The UI display name, or text value, for a search result column. This method is supported only for non-stored select, image, and document fields.
Property	Result.recordType	string (read-only)	Client and server-side scripts	The type of record returned in a search result row.
	Result.id	string (read-only)	Client and server-side scripts	The internal ID for the record returned in a search result row.
	Result.columns	search.Column []	Client and server-side scripts	Array of search.Column objects that encapsulate the columns returned in the search result row.

ResultSet Object Members

The following members are called on [search.ResultSet](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	ResultSet.getRange(options)	search.Result []	Client and server-side scripts	Retrieve a slice of the search result as an array of search.Result objects.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	ResultSet.getRange.promise(options)	search.Result[]	Client scripts	Asynchronously retrieve a slice of the search result as an array of <code>search.Result</code> objects.
	ResultSet.each(callback)	void	Client and server-side scripts	Use a developer-defined function to invoke on each row in the search results, up to 4000 results at a time.
	ResultSet.each.promise(callback)	void	Client scripts	Asynchronously use a developer-defined function to invoke on each row in the search results, up to 4000 results at a time.
Property	ResultSet.columns	search.Column[]	Client and server-side scripts	An array of <code>search.Column</code> objects that represent the columns returned in the search results.

Setting Object Members

The following members are called on `search.Settings`.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	Setting.name	read-only string	Client and server-side scripts	The name of the search parameter.
	Setting.value	read-only string	Client and server-side scripts	The value of the search parameter.

N/search Module Script Samples

The following script samples demonstrate how to use the features of the N/search module.



Important: These samples are designed to run in a NetSuite OneWorld account, so the OneWorld feature must be enabled in your NetSuite account for the samples to work.

Sample 1: Create and run a search



Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a search for customer records. The sample specifies several result columns and one filter, and it logs the first 50 search results.

```

1 /**
2  * @NApiVersion 2.x
3 */

```

```

4
5 require(['N/search'], function(search) {
6     var mySearch = search.create({
7         type: search.Type.CUSTOMER,
8         columns: ['entityid', 'firstname', 'lastname', 'salesrep'],
9         filters: ['entityid', 'contains', 'Adam']
10    });
11
12    var myResultSet = mySearch.run();
13
14    var resultRange = myResultSet.getRange({
15        start: 0,
16        end: 50
17    });
18
19    for (var i = 0; i < resultRange.length; i++) {
20        log.debug(resultRange[i]);
21    }
22 });

```

Sample 2: Create and save a search

i Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a search for sales order records and saves it. The sample specifies several result columns and two filters.

```

1 /**
2 * @NApiVersion 2.x
3 */
4
5 require(['N/search'], function(search) {
6     function createSearch() {
7         var mySalesOrderSearch = search.create({
8             type: search.Type.SALES_ORDER,
9             title: 'My SalesOrder Search',
10            id: 'customsearch_my_so_search',
11            columns: ['entity', 'subsidiary', 'name', 'currency'],
12            filters: [
13                ['mainline', 'is', 'T'],
14                'and', ['subsidiary.name', 'contains', 'CAD']
15            ]
16        });
17
18        mySalesOrderSearch.save();
19    }
20
21    createSearch();
22 });

```

Sample 3: Load and run a search

i Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads and runs a saved search for sales order records. The sample uses the each callback function to process the results.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/search'], function(search) {
6      function loadAndRunSearch() {
7          var mySearch = search.load({
8              id: 'customsearch_my_so_search'
9          });
10
11         mySearch.run().each(function(result) {
12             var entity = result.getValue({
13                 name: 'entity'
14             });
15             var subsidiary = result.getValue({
16                 name: 'subsidiary'
17             });
18
19             return true;
20         });
21     }
22
23     loadAndRunSearch();
24 });

```

Sample 4: Run a search and get a range of result rows

i Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads and runs a saved search for sales order records. The sample obtains the first 100 rows of search results.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/search'], function(search) {
6      function runSearchAndFetchResult() {
7          var mySearch = search.load({
8              id: 'customsearch_my_so_search'
9          });
10
11         var searchResult = mySearch.run().getRange({
12             start: 0,
13             end: 100
14         });
15         for (var i = 0; i < searchResult.length; i++) {
16             var entity = searchResult[i].getValue({
17                 name: 'entity'
18             });
19             var subsidiary = searchResult[i].getValue({
20                 name: 'subsidiary'
21             });
22         }
23     }
24
25     runSearchAndFetchResult();
26 });

```

Sample 5: Run a paginated search

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads and runs a saved search for sales order records. The sample uses the `forEach` callback function to process the paginated results.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/search'], function(search) {
6      function loadAndRunSearch() {
7          var mySearch = search.load({
8              id: 'customsearch_my_so_search'
9          });
10
11         var myPagedData = mySearch.runPaged();
12         myPagedData.pageRanges.forEach(function(pageRange){
13             var myPage = myPagedData.fetch({index: pageRange.index});
14             myPage.data.forEach(function(result){
15                 var entity = result.getValue({
16                     name: 'entity'
17                 });
18                 var subsidiary = result.getValue({
19                     name: 'subsidiary'
20                 });
21             });
22         });
23     }
24
25     loadAndRunSearch();
26 });

```

Sample 6: Search for a custom record type

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a search for a custom record type. To search for a custom record type, you must specify a type of `search.Type.CUSTOM_RECORD` and add the ID of the custom record type (as a string). In this sample, the ID of the custom record type is 6. The custom record also includes a custom field named `custrecord1`.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/search'], function(search) {
6      var myCustomRecordSearch = search.create({
7          type: search.Type.CUSTOM_RECORD + '6';
8          title: 'My Search Title',
9          columns: ['custrecord1']
10     }).run().each(function(result) {
11         // Process each result
12         return true;
13     });

```

14 |});

Sample 7: Search in a custom list



Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a search for items in a custom list. It searches for the internal ID value of an abbreviation in a custom list named `customlist_mylist`.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/search'], function(search) {
6      var internalId = -1;
7      var myCustomListSearch = search.create({
8          type: 'customlist_mylist',
9          columns: [
10              { name: 'internalId' },
11              { name: 'abbreviation' }
12          ]
13      });
14
15      myCustomListSearch.filters = [
16          search.createFilter({
17              name: 'formulatext',
18              formula: '{abbreviation}',
19              operator: search.Operator.IS,
20              values: abbreviation
21          });
22      ];
23
24      var resultSet = myCustomListSearch.run();
25      var results = resultSet.getRange({
26          start: 0,
27          end: 1
28      });
29      for(var i in results) {
30          //log.debug('Found custom list record', results[i]);
31          internalId = results[i].getValue({
32              name:'internalId'
33          });
34      };
35  });

```

Sample 8: Delete a saved search



Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to delete a saved search.

```

1  /**
2   * @NApiVersion 2.x
3   */
4

```

```

5 | require(['N/search'], function(search) {
6 |     function deleteSearch() {
7 |         search.delete({
8 |             id: 'customsearch_my_so_search'
9 |         });
10 |     }
11 |
12 |     deleteSearch();
13 | });

```

search.Search



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a NetSuite search. Use the methods available to search.Search to create a search, run a search, or save a search.
	<p>Note: You do not need to save the search to run it.</p> <p>For a complete list of this object's methods and properties, see Search Object Members.</p>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4     id: 'customsearch_my_so_search'
5 });
6 ...
7 //Add additional code

```

Search.run()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Runs an on-demand search created with search.create(options) or a search loaded with search.load(options), returning the results as a search.ResultSet. Calling this method does not save the search.</p> <p>Use this method with search.create(options) to create and run on-demand searches that are never saved to the database.</p> <p>After you run a search, you can use ResultSet.each(callback) to iterate through the result set and process each result.</p>
---------------------------	---

	<p>Important: When you call this method, consider the following:</p> <ul style="list-style-type: none"> ■ Search result sets are not cached. If records applicable to your search are created, modified, or deleted at the same time you are traversing your result set, your result set may change. ■ For better performance, consider creating a saved search in the UI and loading it in your script using <code>search.load(options)</code> instead of creating the search directly in your script using <code>search.create(options)</code>.
Returns	<code>search.ResultSet</code>
Governance	None
Module	N/search Module
Since	2015.2

Errors

Error Code	Thrown If
<code>SSS_INVALID_SRCH_SETTING</code>	An unknown search parameter name is provided.
<code>SSS_INVALID_SRCH_SETTING_VALUE</code>	An unsupported value is set for the provided search parameter name.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 function loadAndRunSearch() {
4     var mySearch = search.load({
5         id: 'customsearch_my_so_search'
6     });
7     mySearch.run().each(function(result) {
8         var entity = result.getValue({
9             name: 'entity'
10        });
11        var subsidiary = result.getValue({
12            name: 'subsidiary'
13        });
14        return true;
15    });
16 }
```

Search.runPaged(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Runs the current search and returns summary information about paginated results.
---------------------------	--

	<p>Calling this method does not give you the result set or save the search. To retrieve data, use PagedData.fetch(options).</p> <div style="border: 1px solid #f0e68c; padding: 10px; margin-top: 10px;"> ! Important: When you use this method to run a paged search, consider the following: <ul style="list-style-type: none"> ■ Search result sets are not cached. If records applicable to your search are created, modified, or deleted at the same time you are traversing your result set, your result set may change. ■ This method can return a maximum of 1000 pages of search results. </div>
Returns	search.PagedData
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	5 units
Module	N/search Module
Since	2016.1

Parameters

i Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description
options.pageSize	number	optional	Maximum number of entries per page There is an upper limit, a lower limit, and a default setting: <ul style="list-style-type: none"> ■ The maximum number allowed is 1000. ■ The minimum number allowed is 5. ■ By default, the page size is set to 50 entries per page.

Errors

Error Code	Thrown If
SSS_INVALID_SRCH_SETTING	An unknown search parameter name is provided.
SSS_INVALID_SRCH_SETTING_VALUE	An unsupported value is set for the provided search parameter name.

Syntax

! Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/search Module Script Samples .
---	--

```

1 //Add additional code
2 ...
3 var mySearch = search.create({

```

```

4   type: search.Type.CUSTOMER
5 });
6
7 // Run the paged search
8 var pData = mySearch.runPaged({
9   pageSize: 50
10 });
11 ...
12 // Use the count property to count the
13 // search results easily
14 var resultCount = mySearch.runPaged({
15   pageSize: 50
16 }).count;
17 ...
18 //Add additional code

```

Search.runPaged.promise(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Runs the current search asynchronously and returns a search.PagedData Object.
	<p>i Note: The parameters and errors thrown for this method are the same as those for Search.runPaged(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	Search.runPaged(options)
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	5 units
Module	N/search Module
Since	2016.1

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 mySearch.runPaged.promise().then(getPageRangesPromiseChain);
4 ...
5 //Add additional code

```

Search.save()

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Saves a search created by search.create(options) or loaded with search.load(options) . Returns the internal ID of the saved search.
---------------------------	---

	<p>You must set the title and id properties for a new saved search before you save it, either when you create it with search.create(options) or by setting the Search.title and Search.id properties.</p> <p>If you do not set the saved search ID, NetSuite generates one for you. See Search.id.</p>
	<p>Note: You do not need to set these properties if you load a previously saved search with search.load(options) and then save it.</p>
	<p>This method also includes a promise version, Search.save.promise(). For more information about promises, see Promise Object.</p>
Returns	the internal search ID of the saved search as a number
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	5 units
Module	N/search Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	{1}: Missing a required argument: {2}	Required Search.title property not set on search.Search .
NAME_ALREADY_IN_USE	A search has already been saved with that name. Please use a different name.	The Search.title property on search.Search is not unique.
SSS_DUPLICATE_SEARCH_SCRIPT_ID	Saved search script IDs must be unique. Please choose another script ID. If you are trying to modify an existing saved search, use search.load() .	The Search.id property on search.Search is not unique.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 mySalesOrderSearch.save();
4 ...
5 //Add additional code

```

Search.save.promise()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Asynchronously saves a search created by search.create(options) or loaded with search.load(options) . Returns the internal ID of the saved search.
---------------------------	--

	<p>Note: The parameters and errors thrown for this method are the same as those for Search.save(). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	Search.save()
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	5 units
Module	N/search Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 search.create.promise({
4     type: search.Type.SALES_ORDER
5     })
6     .then(function(searchObj) {
7         return searchObj.save.promise()
8     })
9     .then(function (result) {
10         log.debug({
11             details: "Completed: " + result
12         });
13         // do something after completion
14     })
15     .catch(function onRejected(reason) {
16         // do something on rejection
17     });
18 ...
19 //Add additional code

```

Search.searchType



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID name of the record type on which a search is based. Use this if you have the internal ID of the search, but do not know the record type the search was based on. For example, if the search was on a Customer record, this property is customer; if the search was on the Sales Order record type, this property is salesorder.
Type	read-only string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module

Since	2015.2
--------------	--------

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6 log.debug({
7   title: 'record type: ',
8   details: mySearch.searchType
9 });
10 ...
11 //Add additional code

```

Search.searchId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID of the search. The internal ID is available only when the search is either loaded with <code>search.load(options)</code> or after it has been saved with <code>Search.save()</code> . Typical values are 55 or 234 or 87, not a value like customsearch_mysearch. Any ID prefixed with customsearch is a script ID, not the internal system ID for a search.
Type	number (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6 log.debug({
7   title: 'search id #: ',
8   details: mySearch.searchId
9 });
10 ...
11 //Add additional code

```

Search.filters



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>Filters for the search as an array of search.Filter objects. Value is null if the search has no defined filters.</p> <p>You set this value with an array or single search.Filter objects to overwrite any prior filters. Use null to set an empty array and remove any existing filters on this search. Use search.createFilter(options) to create a filter.</p>
	<p> Note: If you want to get or set a search filter expression, use the Search.filterExpression property.</p>
Type	search.Filter[]
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/search Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_SRCH_FILTER	An search filter contains invalid search criteria	Invalid value for search filter type.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3
4 var myFilter = search.createFilter({
5   name: 'entity',
6   operator: search.Operator.ISEMPTY,
7 });
8
9 function createSearch() {
10   var mySalesOrderSearch = search.create({
11     type: search.Type.SALES_ORDER,
12     filters: myFilter
13   });
14 ...
15 //Add additional code

```

Search.filterExpression



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Use filter expressions as a shortcut to create filters (search.Filter).
-----------------------------	---

	<p>A search filter expression is a JavaScript string array of zero or more elements. Each element is one of the following:</p> <ul style="list-style-type: none"> ■ Operator - For a list of supported operators, see search.Operator. ■ Filter term ■ Two or more filter expressions combined logically with 'and', 'or', or 'not' <p>This property accepts nested arrays in which any element of the nested array can be an Object, a string, a number, or a boolean. Use null to set an empty array and remove any existing filter expressions on this search.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: If you want to get or set search filters, use the Search.filters property. </div>
Type	Array[]
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_SRCH_FILTER_EXPR	<p>Malformed search filter expression. This is a general error raised when a filter expression cannot be parsed. For example:</p> <pre>[f1, 'and', 'and', f2]</pre>	The options.filters parameter is not a valid search filter, filter array, or filter expression.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 search.create({
4   type: search.Type.CUSTOMER,
5   filters: [
6     ['email', search.Operator.STARTSWITH, 'kwolff'],
7     'and',
8     [
9       ['id', search.Operator.EQUALTO, 107], 'or',
10      ['id', search.Operator.EQUALTO, 2508]
11    ]
12  ]
13 });
14 ...
15 //Add additional code

```

Search.columns

<p>Note: The content in this help topic pertains to SuiteScript 2.0.</p>	
Property Description	Columns to return for this search as an array of <code>search.Column</code> objects or a string array of column names. You set this value with an array of <code>search.Column</code> objects or a single <code>search.Column</code> to overwrite any prior return columns for the search. Use null to set an empty array and remove any existing columns on this search.
Type	<code>search.Column[] string[]</code>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Errors

Error Code	Thrown If
<code>SSS_INVALID_SRCH_COLUMN</code>	The value passed in was not a string or <code>search.Column</code> Object

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 function createSearch() {
4     var mySalesOrderSearch = search.create({
5         type: search.Type.SALES_ORDER,
6         columns: ['entity', 'subsidiary', 'name', 'currency'],
7     });
8 ...
9 //Add additional code

```

Search.packageId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The application ID for this search.</p> <p>An application ID identifies a SuiteApp project and is a fully qualified name with the following notation:</p> <pre>1 <publisher_id>.<project_id></pre> <p>For example, com.netsuite.mysuiteapp and org.mycompany.helloworld are application IDs.</p> <p>To use this feature, the Show App ID Field preference must be enabled in your NetSuite account. For more information, see the help topic SuiteCloud Development Framework Account Preferences (SDF Developers Only).</p>
-----------------------------	---

Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2019.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var mySalesOrderSearch = search.create({
4   type: search.Type.SALES_ORDER,
5   packageId: 'com.example',
6   columns: ['entity', 'subsidiary', 'name', 'currency']
7 });
8
9 var thePackageId = mySalesOrderSearch.packageId;
10 ...
11 // Add additional code

```

Search.settings

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Search settings for this search as an array of search.Setting objects or a string array of column names. Search settings let you specify search parameters that are typically available only in the UI. You set this value with an array of search.Setting objects or a single search.Setting object. You can create a search.Setting object by calling search.createSetting(options) . You can also set this value with an array of column names, each of which is a string. The supported values for a search.Setting object differ depending on the search parameter that you set. For more information, see Setting.name and Setting.value .
Type	search.Setting [] string[]
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2018.2

Errors

Error Code	Thrown If
SSS_INVALID_SRCH_SETTING	An unknown search parameter name is provided.
SSS_INVALID_SRCH_SETTING_VALUE	An unsupported value is set for the provided search parameter name.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.create({
4     type: 'transaction',
5     columns: [ 'trandate', 'amount', 'entity' ],
6     filters: [
7         search.createFilter({
8             name: 'internalid',
9             operator: search.Operator.ANYOF,
10            values: [13, 12356]
11        }),
12    settings: [
13        search.createSetting({
14            name: 'consolidationtype',
15            value: 'NONE'
16        })
17    ]);
18 ...
19 //Add additional code

```

Search.title



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Title for a saved search. Use this property to set the title for a search before you save it for the first time. You can also set the title for a search when you create it with search.create(options) . The Search.title property is required to save a search with Search.save() .
Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 function createSearch() {
4     var mySalesOrderSearch = search.create({
5         type: search.Type.SALES_ORDER,
6         title: 'My SalesOrder Search',
7         id: 'customsearch_my_so_search',
8     });
9     mySalesOrderSearch.save();

```

```

10 ...
11 //Add additional code

```

Search.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Script ID for a saved search, starting with customsearch. If you do not set this property and then save the search, NetSuite generates a script ID for you.
	<p>Note: This is not the internal NetSuite ID for the saved search. See Search.searchId.</p>
Type	number
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 function createSearch() {
4     var mySalesOrderSearch = search.create({
5         type: search.Type.SALES_ORDER,
6         title: 'My SalesOrder Search',
7         id: 'customsearch_my_so_search',
8     });
9 ...
10 //Add additional code

```

Search.isPublic



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Value is true if the search is public, or false if it is not. By default, all searches created through search.create(options) are private.
Type	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6 mySearch.isPublic = true;
7 ...
8 //Add additional code

```

search.Result



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>Encapsulate a single search result row. Use the methods and properties for search.Result to get the column values for the result row.</p> <p>Note: Use search.ResultSet for the set of results from a search.</p>
	<p>For more information about executing NetSuite searches using SuiteScript, see Searching Overview.</p>
	<p>For a complete list of this object's methods and properties, see Result Object Members.</p>
Supported Script Types	<p>All script types</p>
	<p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6
7 var searchResult = mySearch.run().getRange({
8   start: 0,
9   end: 100
10 });
11 for (var i = 0; i < searchResult.length; i++) {
12   var entity = searchResult[i].getValue({
13     name: 'entity'
14   });
15   var subsidiary = searchResult[i].getValue({
16     name: 'subsidiary'
17   });
18 ...
19 //Add additional code

```

Result.getValue(column)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Used on formula and non-formula (standard) fields. Returns the value of a specified search result column. For convenience, this method takes a single search.Column Object.</p> <p>Note: This method is overloaded. You can also use Result.getValue(options) to get column values based on the name, join, and summary values for a column.</p>
Returns	<p>The return type depends on the type of search result column that was specified:</p> <ul style="list-style-type: none"> ■ boolean if the column is a check box field ■ number if the column is a record, list, decimal number, or image field, with the following considerations: <ul style="list-style-type: none"> □ For image fields, the returned number represents the ID of the image file. ■ string for all other column types, with the following considerations: <ul style="list-style-type: none"> □ For multiselect fields, the returned string represents a comma-separated list of IDs. Each ID represents a selectable option in the field. □ For date/time fields, the returned string represents the formatted string value of the date. You can use methods in the N/format module to work with this string (for example, converting it to a Date object). For more information, see N/format Module.
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	<p>None</p>
Module	N/search Module
Since	<p>2015.2</p>

Parameters

Parameter	Type	Required / Optional	Description
column	search.Column	Required	The search result column from which to return a value.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6
7 var resultSet = mySearch.run();
8 var firstResult = resultSet.getRange({
9   start: 0,
10  end: 1
11 });
12
13 // get the value of the second column (zero-based index)

```

```

14 |     var value = firstResult.getValue(resultSet.columns[1]);
15 |
16 |     log.debug({
17 |         title: 'Value:',
18 |         details: value
19 |     });
20 |
21 | //Add additional code

```

Result.getValue(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Used on formula and non-formula (standard) fields. Returns the value of a specified search result column. Takes in arguments for name, join, and summary.</p> <p>i Note: This method is overloaded. You can also use Result.getValue(column) to get column values. This method takes in a single search.Column.</p> <p>⚠ Important: If you have multiple search return columns and you apply grouping, all columns must include a summary property.</p>
Returns	<p>The return type depends on the type of search result column that was specified:</p> <ul style="list-style-type: none"> ■ boolean if the column is a check box field ■ number if the column is a record, list, decimal number, or image field, with the following considerations: <ul style="list-style-type: none"> □ For image fields, the returned number represents the ID of the image file. ■ string for all other column types, with the following considerations: <ul style="list-style-type: none"> □ For multiselect fields, the returned string represents a comma-separated list of IDs. Each ID represents a selectable option in the field. □ For date/time fields, the returned string represents the formatted string value of the date. You can use methods in the N/format module to work with this string (for example, converting it to a Date object). For more information, see N/format Module.
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/search Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	Required	The search return column name.
options.join	string	Optional	The join id for this search return column.

Parameter	Type	Required / Optional	Description
			Join IDs are listed in the Records Browser. For more information, see the help topic Working with the SuiteScript Records Browser .
options.summary	string	Optional	The summary type for this column. See search.Summary .
options.func	string	Optional	Special function for the search column. See Column.function .

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var searchResults = mySearch.run().getRange({
4   start: 0,
5   end: 100
6 });
7 for (var i = 0; i < searchResults.length; i++) {
8   var amount = searchResults[i].getValue({
9     name: 'amount'
10 });
11   var entity = searchResults[i].getValue({
12     name: 'name',
13     join: 'location'
14 });
15 ...
16 //Add additional code

```

Result.getText(column)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Used on select, image, and document fields. Returns the text value of a specified search result column. For convenience, this method takes a single search.Column Object.
	i Note: This method is overloaded. You can also use Result.getText(options) to get column text value based on the name, join and summary values for a column.
Returns	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/search Module
Since	2015.2

Parameters

Parameter	Type	Required / Optional	Description
column	search.Column	Required	Name of the search result column.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6
7 var resultSet = mySearch.run();
8 var firstResult = resultSet.getRange({
9   start: 0,
10  end: 1
11 })[0];
12
13 // get the text value of the second column (zero-based index)
14 var value = firstResult.getText(resultSet.columns[1]);
15
16 log.debug({
17   title: 'Value: ',
18   details: value
19 });
20 ...
21 //Add additional code

```

Result.getText(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Used on select, image, and document fields. Returns the text value of a specified search result column. Note: This method is overloaded. You can also use Result.getText(column) to get a column value. This method takes in a single search.Column .
Returns	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/search Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	Required	The name of the search column.

Parameter	Type	Required / Optional	Description
options.join	string	Optional	The join internal ID for the search column.
options.summary	string	Optional	The summary type used for the search column. See search.Summary .

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var searchResults = mySearch.run().getRange({
4   start: 0,
5   end: 100
6 });
7 for (var i = 0; i < searchResults.length; i++) {
8   var amount = searchResults[i].getText({
9     name: 'amount'
10 });
11 var entity = searchResults[i].getText({
12   name: 'name',
13   join: 'location'
14 });
15 ...
16 //Add additional code

```

Result.recordType



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The type of record returned in a search result row.
Type	search.Type enum
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6
7 var searchResult = mySearch.run();

```

```

8 log.debug({
9   title: 'Record Type: ',
10  details: searchResult.recordType
11 });
12 ...
13 //Add additional code

```

Result.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID for the record returned in a search result row. This ID is a number, but it is stored in this property as a string (for example, '237').
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var mySearch = search.create({
4   type: search.Type.CUSTOMER
5 });
6
7 var resultSet = mySearch.run();
8
9 resultSet.each(function(result) {
10   log.debug({
11     title: 'Record Internal ID: ',
12     details: result.id
13   });
14
15   return true;
16 });
17 ...
18 //Add additional code

```

Result.columns



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Array of Search.Column objects that encapsulate the columns returned in the search result row.
Type	search.Column[] (read-only)

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6
7 var firstResult = mySearch.run().getRange({
8   start: 0,
9   end: 1
10 })[0];
11 log.debug({
12   details: "There are " + firstResult.columns.length + " columns in the result."
13 });
14
15 firstResult.columns.forEach(function(col){ //log each column
16   log.debug({
17     details: col
18   });
19 });
20 ...
21 //Add additional code

```

search.Column

i Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a single search column in a <code>search.Search</code> . Use the methods and properties available to the Column object to get or set Column properties. You create a search column object with <code>search.createColumn(options)</code> and add it to a <code>search.Search</code> object that you create with <code>search.create(options)</code> or load with <code>search.load(options)</code> . You can pass a Column object as a parameter to the <code>Result.getValue(column)</code> or <code>Result.getText(column)</code> methods. In addition, <code>search.ResultSet</code> contains an array of Column objects returned in the results of a search. For a complete list of this object's methods and properties, see Column Object Members .
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 search.create({
4     type: search.Type.TRANSACTION,
5     columns: [
6         'trandate',
7         'amount',
8         'entity',
9         'entity.firstname',
10        'entity.email',
11        search.createColumn({
12            name: 'formulatext',
13            formula: "{lastname}||', '||{firstname}"
14        })
15    ],
16
17    // When the search is executed, the corresponding column in the result will then contain a value in the form: Last Name, First Name
18 ...
19 //Add additional code

```

Column.setWhenOrderedBy(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the search column for which the minimal or maximal value should be found when returning the search.Column value. For example, can be set to find the most recent or earliest date, or the largest or smallest amount for a record, and then the search.Column value for that record is returned. <div style="background-color: #e0f2ff; padding: 10px;"> i Note: You can only use this method if you use MIN or MAX as the summary type on a search column with the Result.getValue(options) method. </div>
Returns	search.Column
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/search Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	Required	The name of the search column for which the minimal or maximal value should be found.

Parameter	Type	Required / Optional	Description
options.join	string	Required	The join id for the search column.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // Execute a customer search that returns the amount of the most recent sales order per customer
4
5 var filters = [];
6 var columns = [];
7
8 filters[0] = search.createFilter({
9   name: 'recordtype',
10  join: 'transaction',
11  operator: search.Operator.IS,
12  values: 'salesorder'
13 });
14 filters[1] = search.createFilter({
15   name: 'mainline',
16   join: 'transaction',
17   operator: search.Operator.IS,
18   values: true
19 });
20
21 columns[0] = search.createColumn({
22   name: 'entityid',
23   summary: search.Summary.GROUP
24 });
25 columns[1] = search.createColumn({
26   name: 'totalamount',
27   join: 'transaction',
28   summary: search.Summary.MAX
29 });
30
31 columns[1].setWhenOrderedBy({
32   name: 'trandate',
33   join: 'transaction'
34 });
35
36 var mySearch = search.create({
37   type: 'customer',
38   filters: filters,
39   columns: columns
40 });
41 var resultsArray = mySearch.run().getRange({
42   start: 0,
43   end: 100
44 });
45 ...
46 // Add additional code

```

Column.name

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Name of a search column as a string.
Type	string (read-only)

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // Create a search definition that includes search columns
4 var mySearch = search.create({
5   type: search.Type.CUSTOMER,
6   columns: [
7     search.createColumn({
8       name: 'entityid'
9     }),
10    search.createColumn({
11      name: 'email'
12    })
13  ]
14 });
15
16 // Retrieve the first search column and log its name
17 var myColumn = mySearch.columns[0];
18 log.debug(myColumn.name);
19
20 // Run the search
21 var results = mySearch.run();
22 ...
23 // Add additional code

```

Column.join

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Join ID for a search column as a string.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // Create a joined column. The name parameter is a field name on

```

```

4 // the joined record type, and the join parameter is a field name
5 // on the base record type that references the joined record type.
6 //
7 // In this example, a search is created for customer records.
8 // Customer records include a 'salesrep' field, which references an
9 // employee record. This joined column represents the value of the
10 // 'firstname' field on the referenced employee record.
11 var myColumn = search.createColumn({
12     name: 'firstname',
13     join: 'salesrep'
14 });
15
16 var mySearch = search.create({
17     type: search.Type.CUSTOMER,
18     columns: [
19         'entityid',
20         'email',
21         myColumn
22     ]
23 });
24
25 var theJoin = myColumn.join;
26 ...
27 // Add additional code

```

Column.summary

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the summary type for a search column.
Type	search.Summary enum
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

! **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 log.debug({
4     details: 'Summary Type for Search Column: '
5         + columnObj.summary
6     });
7 ...
8 //Add additional code

```

Column.formula

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Formula used for a search column as a string.
-----------------------------	---

	To set this value, you must use formulatext, formulanumeric, formuladatetime, formulapercents, or formulacurrency.
Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

For example, in the UI, a field with a custom UI label named **Customer Name** is set by a formula of type **Formula (Text)** and the formula is defined with the following formula:

```

1 //Add additional code
2 ...
3 var columnObj = search.createColumn({
4   name: 'formulatext',
5   formula: "{firstname} || ' ' || {lastname}"
6 });
7 ...
8 //Add additional code

```

In the above formula, `firstname` and `lastname` are script IDs for the fields on the Customer record form.

Column.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Label used for the search column. You can only get or set custom labels with this property.
Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var columnObj = search.createColumn({
4   name: 'formulanumeric',
5   label: 'Numeric Formula'

```

```

6  });
7 ...
8 //Add additional code

```

Column.function



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Special function applied to values in a search column. See Supported Functions .
Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Supported Functions

The following table lists the supported functions and their internal IDs:

Internal ID	Name	Date Function	Output
percentOfTotal	% of Total	No	percent
absoluteValue	Absolute Value	No	integer
ageInDays	Age In Days	Yes	integer
ageInHours	Age In Hours	Yes	integer
ageInMonths	Age In Months	Yes	integer
ageInWeeks	Age In Weeks	Yes	integer
ageInYears	Age In Years	Yes	integer
calendarWeek	Calendar Week	Yes	date
day	Day	Yes	date
month	Month	Yes	text
negate	Negate	No	integer
numberAsTime	Number as Time	No	text
quarter	Quarter	Yes	text
rank	Rank	No	integer
round	Round	No	float
roundToHundredths	Round to Hundredths	No	float
roundToTenths	Round to Tenths	No	float
weekOfYear	Week of Year	Yes	text

Internal ID	Name	Date Function	Output
year	Year	Yes	text

Errors

Error Code	Message	Thrown If
INVALID_SRCH_FUNCTN	A search.Column contains an invalid function: {1}.	Unknown function is set.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var columnObj = search.createColumn({ //the age of the sales order in days
4   name: 'trandate',
5   function: 'ageInDays'
6 });
7 ...
8 //Add additional code

```

Column.sort

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The sort order of the column. Use search.createColumn(options) and a value from the search.Sort enum to set the value of this property. If Column.sort is not set, the column is not sorted in any particular order. After you create a column, you cannot change the sort order of the column. If you use the same column in another search and specify a new sort order, the previous sort order is still used.
Type	search.Sort enum
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var columnObj = search.createColumn({
4   name: 'invoice',

```

```

5   sort: search.Sort.DESC
6 });
7 ...
8 //Add additional code

```

search.Filter

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a search filter used in a search. Use the properties for the Filter object to get and set the filter properties. You create a search filter object with search.createFilter(options) and add it to a search.Search object that you create with search.create(options) or load with search.load(options) . Note: NetSuite uses an implicit AND operator with search filters, as opposed to filter expressions which explicitly use either AND and OR operators.
	Use the following guidelines with the Filter object: <ul style="list-style-type: none">■ To search for a "none of null" value, meaning do not show results without a value for the specified field, use a value of @NONE@ in the Filter.formula property.■ To search on checkbox fields, use the IS operator with a value of T or F to search for checked or unchecked fields, respectively. For a complete list of this object's methods and properties, see Filter Object Members .
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearchFilter = search.createFilter({
4   name: 'entity',
5   operator: search.Operator.ISEMPTY,
6 });
7 ...
8 //Add additional code

```

Filter.name

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Name or internal ID of the search field as a string.
-----------------------------	--

	For more information, see search.createFilter(options) .
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 log.debug({
4     details: 'Filter Name: '
5         + filterObj.name
6 });
7 ...
8 //Add additional code

```

Filter.join

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Join ID for the search filter as a string.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // Create a filter joined to another record type. When you create a joined filter:
4 // - The name property is the field ID of the field in the joined record that you are filtering on
5 // - The join property is the field ID of the field in the current record that contains the record
6 // type you want to join to
7 // - The operator property is the operator to use to filter the results
8 // - The values property contains the values to use to filter the results
9 search.createFilter({
10     name: 'joined_record_field_id'
11     join: 'current_record_field_id',
12     operator: search.Operator.IS,

```

```

13     values: ['valueToFilter']
14 });
15 ...
16 // Add additional code

```

Filter.operator



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Operator used for the search filter. This value is set with the search.Operator enum. The search.Operator enum contains the valid operator values for this property.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearchFilter = search.createFilter({
4     name: 'entity',
5     operator: search.Operator.ISEMPTY
6 });
7 log.debug({
8     details: 'Operator Used: '
9         + mySearchFilter.operator
10 });
11 ...
12 //Add additional code

```

Filter.summary



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Summary type for the search filter. Use this property to get or set the value of the summary type. See search.Summary .
Type	search.Summary
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_SRCH_FILTER_SUM	A search.Filter contains an invalid summary type: {1}.	Unknown summary type is set.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearchFilter = search.createFilter({
4   name: 'entity',
5   operator: search.Operator.ISNOTEMPTY,
6   summary: search.Summary.GROUP
7 });
8 ...
9
10 //Add additional code

```

Filter.formula

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Formula used by the search filter. Use this property to get or set the formula used by the search filter. For more information about the formula property, see search.createFilter(options) . For more information about using formulas in searches, see the help topic Formulas in Search .
Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

For more examples of search formulas, see the help topic [Search Formula Examples and Tips](#).

```

1 // Add additional code
2 ...
3 // myFilter is a search.Filter object created using search.createFilter(options)
4 // Get the filter formula
5 var theFilterFormula = myFilter.formula;
6
7 // Set the filter formula

```

```

8 | myFilter.formula = '{contribution} - {partnercontribution}';
9 |
10 | // Add additional code

```

search.ResultSet



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>Encapsulates a set of search results returned by Search.run().</p> <p>Use the methods and properties for the ResultSet object to iterate through each result returned by the search or access an arbitrary slice of results. The maximum number of results in a ResultSet object is 4000. If a search matches more than 4000 results, you must use Search.runPaged(options) or Search.runPaged.promise(options) to retrieve the full set of results.</p> <p>For a complete list of this object's methods and properties, see ResultSet Object Members.</p> <div style="background-color: #ffffcc; border: 1px solid #ffcc00; padding: 5px;"> <p>Important: Search result sets are not cached. If records applicable to your search are created, modified, or deleted at the same time you are traversing your result set, your result set may change.</p> </div>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // Load a saved search. Alternatively, you can create a search using search.create(options)
4 // and other methods in the N/search module.
5 var mySearch = search.load({
6   id: 'customsearch_my_cs_search'
7 });
8
9 // Run the search, and use ResultSet.each(callback) to define a callback function to
10 // execute on each search result.
11 //
12 // In this example, the saved search that was loaded above searches for Customer records.
13 // The Result.getValue(options) method obtains the search result value of one of the search
14 // columns that was specified in the search definition. Both 'entityid' and 'email' are valid
15 // search column names for a Customer record.
16 mySearch.run().each(function(result) {
17   var entity = result.getValue({
18     name: 'entityid'
19   });
20   log.debug(entity);
21
22   var email = result.getValue({
23     name: 'email'
24   });
25   log.debug(email);
26

```

```

27     return true;
28 });
29 ...
30 // Add additional code

```

ResultSet.getRange(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Retrieve a slice of the search result as an array of search.Result objects.</p> <p>The start parameter is the inclusive index of the first result to return. The end parameter is the exclusive index of the last result to return. For example, <code>getRange(0, 10)</code> retrieves 10 search results, at index 0 through index 9. Unlimited rows in the result are supported, however you can only return 1,000 at a time based on the index values.</p> <p>If there are fewer results available than requested, then the array will contain fewer than end - start entries. For example, if there are only 25 search results, then <code>getRange(20, 30)</code> will return an array of 5 search.Result objects.</p> <p>If you specify a range for which there are no results, an empty array is returned. For example, if there are 25 search results, then <code>getRange(30, 40)</code> will return an empty array.</p>
Returns	search.Result[]
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	10 units
Module	N/search Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.start</code>	number	Required	Index number of the first result to return, inclusive.
<code>options.end</code>	number	Required	Index number of the last result to return, exclusive.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var results = rs.getRange({
4   start: 0,
5   end: 1000
6 });
7 ...
8 //Add additional code

```

ResultSet.getRange.promise(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to asynchronously retrieve a slice of the search result as an array of search.Result objects.
	<p>Note: For information about the parameters and errors thrown for this method, see ResultSet.getRange(options). For additional information on promises, see Promise Object.</p>
Returns	search.Result[]
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/search Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 var results = rs.getRange.promise({
4     start: 0,
5     end: 1000
6 });
7     .then(function(response){
8         log.debug({
9             title: 'Completed',
10            details: response
11        });
12    })
13    .catch(function onRejected(reason) {
14        log.debug({
15            title: 'Failed: ',
16            details: reason
17        });
18    })
19 ...
20
21 //Add additional code

```

ResultSet.each(callback)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Use a developer-defined function to invoke on each row in the search results, up to 4000 results at a time. The callback function must use the following signature: boolean callback(result.Result result);
---------------------------	--

	The callback function takes a search.Result object as an input parameter and returns a boolean which can be used to stop the iteration with a value of false, or continue the iteration with a value of true.
	<p>Important: The work done in the context of the callback function counts towards the governance of the script that called it. For example, if the callback function is running in the context of a scheduled script, which has a 10,000 unit governance limit, make sure the amount of processing within the callback function does not put the entire script at risk of exceeding scheduled script governance limits.</p>
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/search Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
callback	function	Required	Named JavaScript function or anonymous inline function that contains the logic to process a search.Result object.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 mySearch.run().each(function(result) {
4     var entity = result.getValue({
5         name: 'entity'
6     });
7     var subsidiary = result.getValue({
8         name: 'subsidiary'
9     });
10    return true;
11 });
12 ...
13 //Add additional code

```

ResultSet.each.promise(callback)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Asynchronously uses a developer-defined function to invoke on each row in the search results, up to 4000 results at a time. The callback function must use the following signature:
---------------------------	---

	boolean callback(result.Result result);
	<p>Note: The parameters and errors thrown for this method are the same as those for ResultSet.each(callback). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/search Module
Since	2015.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 mySearch.run().each.promise(function(result) {
4     var entity = result.getValue({
5         name: 'entity'
6     });
7     var subsidiary = result.getValue({
8         name: 'subsidiary'
9     });
10    return true;
11 })
12 .then(function(response){
13     log.debug({
14         title: 'Completed',
15         details: response
16     });
17 })
18 .catch(function onRejected(reason) {
19     log.debug({
20         title: 'Failed: ',
21         details: reason
22     });
23 })
24 ...
25 //Add additional code

```

ResultSet.columns

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An array of search.Column objects that represent the columns returned in the search results.
Type	search.Column[] This property is read-only

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6
7 var resultSet = mySearch.run();
8 log.debug({
9   details: "There are " + resultSet.columns.length + " columns in the result set:"
10 });
11
12 resultSet.columns.forEach(function(col){ //log each column
13   log.debug({
14     details: col
15   });
16 });
17 ...
18 //Add additional code

```

search.Page

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates an individual search page containing a result set for a paginated search. For a complete list of this object's methods and properties, see Page Object Members .
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	Version 2015 Release 1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```
1 //Add additional code
```

```

2 ...
3 var page = pagedData.fetch({
4   index: lastPageRange.index
5 });
6 ...
7 //Add additional code

```

Page.next()

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to fetch the next segment of data (bounded by search.PageRange). Moves the current page to next range.
Returns	Void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	5 units
Module	N/search Module
Since	2016.1

Errors

Error Code	Message	Thrown If
INVALID_PAGE_RANGE	Invalid page range.	The page range is invalid, or when the page is the last page.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 while (!page.isFirst){
4   page = page.next();
5 ...
6 //Add additional code

```

Page.next.promise()

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to asynchronously fetch the next segment of data (bounded by search.PageRange). Moves the current page to another range. The promise is complete when the data for this range is loaded or rejected.
---------------------------	--

	Note: The parameters and errors thrown for this method are the same as those for Page.next() . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	Page.next()
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	5 units
Module	N/search Module
Since	2016.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...
3 // In this snippet, myPage is a Page object that encapsulates a page of search results,
4 // and processPage is the name of a callback function to execute when the promise
5 // method returns
6 return myPage.next.promise().then(processPage);
7 ...
8 // Add additional code

```

Page.prev()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Method used to fetch the previous segment of data (bounded by search.PageRange). Moves the current page to previous range.
Returns	Void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	5 units
Module	N/search Module
Since	2016.1

Errors

Error Code	Message	Thrown If
INVALID_PAGE_RANGE	Invalid page range.	The page range is invalid, or when the page is the first page.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 while (!page.isLast){
4     page = page.prev();
5 ...
6 //Add additional code

```

Page.prev.promise()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Method used to asynchronously fetch the previous segment of data (bounded by search.PageRange).</p> <p>Moves the current page to another range. The promise is complete when the data for this range is loaded or rejected.</p> <p>Note: The parameters and errors thrown for this method are the same as those for Page.prev(). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	Page.prev()
Supported Script Types	<p>All client-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Client Script Type.</p>
Governance	5 units
Module	N/search Module
Since	2016.1

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 return mypage.prev.promise().then(processPage);
4 ...
5 //Add additional code

```

Page.data



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The results from a paginated search.
-----------------------------	--------------------------------------

Type	search.Result[] This property is read-only.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 function processPage(page){
4     page.data.forEach(function(value){
5         log.debug({
6             details: "data: " + page.data
7         });
8     ...
9 //Add additional code

```

Page.isFirst



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the page is within the first range of the result set. Flags the start of the data collection.
Type	boolean true false This property is read-only.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 while (!page.isFirst){
4     page = page.next();
5 ...
6 //Add additional code

```

Page.isLast



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether a page is within the last range of the result set. Flags the end of the data collection.
Type	boolean true false This property is read-only.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 while (!page.isLast){
4     page = page.prev();
5 ...
6 //Add additional code

```

Page.pagedData



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The PagedData Object used to fetch this Page Object.
Type	search.PagedData This property is read-only.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var lastPageRange = pagedData.pageRanges[pagedData.pageRanges.length - 1];
4 ...

```

```
5 | //Add additional code
```

Page.pageRange

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The PageRange Object used to fetch this Page Object. Page boundary information with the key and label.
Type	search.PageRange This property is read-only.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```
1 //Add additional code
2 ...
3 log.debug({
4   details: "Page Range: " + mySearchPage.pageRange
5 });
6 ...
7
8 //Add additional code
```

search.PagedData

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Holds metadata for a paginated query. This object provides a high-level view of a search result, giving the total count of records, a list of pages ranges, and page size. For paged searches, the maximum number of result rows per page is 1000. The minimum number of result rows per page is 5, except for the last page in the result set (because the last page may include fewer than 5 results). For a complete list of this object's methods and properties, see PagedData Object Members . Important: Search result sets are not cached. If records applicable to your search are created, modified, or deleted at the same time you are traversing your result set, your result set may change.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module

Since	Version 2015 Release 1
--------------	------------------------

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 // Run the paged search
4 var pagedData = mySearch.runPaged({
5   pageSize: 1000
6 });
7 ...
8 // Use the count property to count the
9 // search results easily
10 var resultCount = mySearch.runPaged({
11   pageSize: 1000
12 }).count;
13 ...
14 //Add additional code

```

PagedData.fetch(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	This method retrieves the data within the specified page range. This method also includes a promise version, <code>PagedData.fetch.promise()</code> . For more information about promises, see Promise Object .
Returns	<code>search.Page</code>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	5 units
Module	N/search Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>pageRange.index</code>	number	required	The index of the page range that bounds the desired data. Page range indexes start at 0.

Errors

Error Code	Message	Thrown If
<code>INVALID_PAGE_RANGE</code>	Invalid page range.	The page range is not valid.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var mySearch = search.create({
4   type: search.Type.CUSTOMER,
5   columns: [
6     'entityid',
7     'email'
8   ]
9 });
10
11 var myPagedResults = mySearch.runPaged({
12   pageSize: 10
13 });
14
15 var thePageRanges = myPagedResults.pageRanges;
16
17 // Use PagedData.fetch(options) to retrieve a page of results (as a
18 // search.Page object) by index
19 var theData = myPagedResults.fetch({
20   index: 5
21 });
22 ...
23 // Add additional code

```

PagedData.fetch.promise()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	This method asynchronously retrieves the data bounded by the pageRange parameter.
	<p>Note: The parameters and errors thrown for this method are the same as those for PagedData.fetch(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	PagedData.fetch(options)
Supported Script Types	All client-side scripts For more information see, SuiteScript 2.0 Client Script Type .
Governance	5 units
Module	N/search Module
Since	2016.1

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 // Add additional code
2 ...

```

```

3 | return pagedData.fetch.promise().then(processPage);
4 | ...
5 | // Add additional code

```

PagedData.count



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The total number of results when Search.runPaged(options) was executed.
Type	number This property is read-only.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 | // Add additional code
2 |
3 | var mySearch = search.create({
4 |   type: search.Type.CUSTOMER,
5 |   columns: [
6 |     'entityid',
7 |     'email'
8 |   ]
9 | });
10 |
11 | var myPagedResults = mySearch.runPaged({
12 |   pageSize: 10
13 | });
14 |
15 | var theCount = myPagedResults.count;
16 |
17 | // Add additional code

```

PagedData.pageRanges



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The collection of search.PageRange objects that divide the entire result set into smaller groups. This property includes page range information with the key and label for rendering. Page range indexes start at 0.
Type	search.PageRange[] This property is read-only.
Supported Script Types	All script types

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var mySearch = search.create({
4   type: search.Type.CUSTOMER,
5   columns: [
6     'entityid',
7     'email'
8   ]
9 });
10
11 var myPagedResults = mySearch.runPaged({
12   pageSize: 10
13 });
14
15 var thePageRanges = myPagedResults.pageRanges;
16
17 // Use PagedData.fetch(options) to retrieve a page of results (as a
18 // search.Page object) by index
19 var theData = myPagedResults.fetch({
20   index: 5
21 });
22 ...
23 // Add additional code

```

PagedData.pageSize

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Maximum number of entries per page Possible values are 5 - 1000 entries per page.
Type	number This is a read-only property.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```
1 // Add additional code
```

```

2 ...
3 var mySearch = search.create({
4   type: search.Type.CUSTOMER,
5   columns: [
6     'entityid',
7     'email'
8   ]
9 });
10
11 var myPagedResults = mySearch.runPaged({
12   pageSize: 10
13 });
14
15 // In this example, the value of myPagedResults.pageSize is 10
16 var thePageSize = myPagedResults.pageSize;
17
18 // Use PagedData.fetch(options) to retrieve a page of results (as a
19 // search.Page object) by index
20 var theData = myPagedResults.fetch({
21   index: 5
22 });
23 ...
24 // Add additional code

```

PagedData.searchDefinition

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The search criteria used to execute the result set for this PagedData Object.
Type	read-only search.Search
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 log.debug({
4   details: "Search Details: " + myPagedData.searchDefinition
5 });
6 ...
7 //Add additional code

```

search.PageRange

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	Defines the page range to contain the result set For a complete list of this object's properties, see PageRange Object Members .
---------------------------	---

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	Version 2015 Release 1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var page = pagedData.fetch({
4   index: lastPageRange
5 });
6 ...
7 //Add additional code

```

PageRange.compoundLabel

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Human-readable label with beginning and ending range identifiers
Type	read-only string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 log.debug({
4   details: "Page Range Description: " + myPageRange.compoundLabel
5 });
6 ...
7 //Add additional code

```

PageRange.index

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The index of the page range.
-----------------------------	------------------------------

	Page range indexes start at 0.
Type	<p>number</p> <p>This property is read-only.</p>
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/search Module
Since	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 log.debug({
4   details: "Page Range Index: " + myPageRange.index
5 });
6 ...
7
8 //Add additional code

```

search.setting



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>Defines a search setting.</p> <p>Search settings let you specify search parameters that are typically available only in the UI. The following settings are supported:</p> <ul style="list-style-type: none"> ■ Consolidated Exchange Rate: This setting affects how consolidation is performed (for example, consolidation using the Average rate type, consolidation using the Historical rate type, and so on). This setting applies to transaction searches, and it is applicable only to OneWorld accounts. ■ Show Period End Transactions: This setting indicates whether period end transactions are included in search results. This setting applies to transaction searches, and it is applicable only to OneWorld accounts. It also requires the Show Period End transactions feature to be enabled. <p>Use search.createSetting(options) to create a setting. After you create your settings, assign them as array values to Search.settings.</p> <p>For a complete list of this object's properties, see Setting Object Members.</p>
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/search Module
Since	2018.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.create({
4     type: 'transaction',
5     columns: [ 'trandate', 'amount', 'entity' ],
6     filters: [
7         search.createFilter({
8             name: 'internalid',
9             operator: search.Operator.ANYOF,
10            values: [13, 12356]
11        }),
12    settings: [
13        search.createSetting({
14            name: 'consolidationtype',
15            value: 'NONE'
16        })
17    ]
18 });
19 //Add additional code

```

Setting.name

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of the search parameter. This property is set when you call search.createSetting(options) . The following values are supported for this property: <ul style="list-style-type: none"> ■ consolidationtype: This value corresponds to the Consolidated Exchange Rate setting. ■ includeperiodendtransactions: This value corresponds to the Show Period End Transactions setting.
Type	string This property is read-only.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2018.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.create({
4     type: 'transaction',
5     columns: [ 'trandate', 'amount', 'entity' ],
6     filters: [

```

```

7   search.createFilter({
8     name: 'internalid',
9     operator: search.Operator.ANYOF,
10    values: [13, 12356]
11  }),
12  settings: [
13    search.createSetting({
14      name: 'consolidationtype',
15      value: 'NONE'
16    })
17  });
18 ...
19 //Add additional code

```

Setting.value

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>The value of the search parameter.</p> <p>This property is set when you call <code>search.createSetting(options)</code>. If you specify <code>consolidationtype</code> as the search parameter name (<code>Setting.name</code>), the following values are supported for this parameter:</p> <ul style="list-style-type: none"> ■ ACCTTYPE ■ AVERAGE ■ CURRENT ■ HISTORICAL ■ NONE <p>If you specify <code>includeperiodendtransactions</code> as the search parameter name (<code>Setting.name</code>), the following values are supported for this parameter:</p> <ul style="list-style-type: none"> ■ F ■ FALSE ■ T ■ TRUE <p>These values are not case sensitive.</p>
Type	<p>string</p> <p>This property is read-only.</p>
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	<p>N/search Module</p>
Since	<p>2018.2</p>

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.create({

```

```

4   type: 'transaction',
5   columns: [ 'trandate', 'amount', 'entity' ],
6   filters: [
7     search.createFilter({
8       name: 'internalid',
9       operator: search.Operator.ANYOF,
10      values: [13, 12356]
11    }),
12   settings: [
13     search.createSetting({
14       name: 'consolidationtype',
15       value: 'NONE'
16     })
17   });
18 ...
19 //Add additional code

```

search.create(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a new search and returns it as a search.Search object.</p> <p>The search can be modified and run as an on demand search with Search.run(), without saving it. Alternatively, calling Search.save() will save the search to the database, so it can be reused later in the UI or loaded with search.load(options).</p> <p>Note: This method is agnostic in terms of its options.filters argument. It can accept input of a single search.Filter object, an array of search.Filter objects, or a search filter expression.</p> <p>The <code>search.create(options)</code> method also includes a promise version, <code>search.create.promise(options)</code>. For more information about promises, see Promise Object.</p> <p>Important: When you use this method to create a search, consider the following:</p> <ul style="list-style-type: none"> ■ When you define the search, make sure you sort using the field with the most unique values, or sort using multiple fields. Sorting with a single field that has multiple identical values can cause the result rows to be in a different order each time the search is run. ■ You cannot directly create a filter or column for a list/record type field in SuiteScript by passing in its text value. You must use the field's internal ID. If you must use the field's text value, you can create a filter or column with a formula using name: 'formulatext'.
Returns	search.Search
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/search Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.type	string	Required	The search type that you want to base the search on. Use the search.Type enum for this argument.	2015.2
options.filters	search.Filter search.Filter[] Object Object[] string string[]	Optional	<p>A single search.Filter object, an array of search.Filter objects, a search filter expression, or an array of search filter expressions.</p> <p>A search filter expression can be passed in as an Object with the following properties:</p> <ul style="list-style-type: none"> ▪ name (required) ▪ join ▪ operator (required) ▪ summary ▪ formula ▪ values <p>For more information about these properties, see Filter Object Members.</p> <p>If a provided filter value has an incorrect type (for example, a string instead of a number), the filter value is ignored. For server-side scripts, a log entry is created when an incorrect type is provided.</p> <p>Note: You can further filter the returned search.Search object by adding additional filters with Search.filters or Search.filterExpression.</p>	2015.2
options.filterExpression	Object[]	Optional	<p>Search filter expression for the search as an array of expression objects.</p> <p>A search filter expression is a JavaScript string array of zero or more elements. Each element is one of the following:</p> <ul style="list-style-type: none"> ▪ Operator - either 'NOT', 'AND', or 'OR' ▪ Filter term ▪ Nested search filter expression <p>You set this value with an array of expression objects or single filter expression object to overwrite any prior filter expressions. Use null to set an empty array and remove any existing filter expressions on this search.</p> <p>Note: If you want to get or set a search filters, use the Search.filters property.</p> <p>This parameter sets the value for the Search.filterExpression property.</p>	2016.2

Parameter	Type	Required / Optional	Description	Since
options.columns	search.Column search.Column[] Object Object[] string string[]	Optional	<p>A single search.Column object or array of search.Column objects.</p> <p>You can optionally pass in an Object or array of Objects with the following properties to represent a Column:</p> <ul style="list-style-type: none"> ■ name (required) ■ formula ■ function ■ join ■ label ■ sort ■ summary <p>For more information about these properties, see Column Object Members.</p>	2015.2
options.packageId	string	Optional	The application ID for this search.	2019.2
options.settings	search.Setting search.Setting[] Object Object[] string string[]	Optional	<p>Search settings for this search as a single search.Setting object or an array of search.Setting objects. Search settings let you specify search parameters that are typically available only in the UI. See Search.settings.</p> <p>You can optionally pass in an Object or array of Objects with the following properties to represent a setting:</p> <ul style="list-style-type: none"> ■ name ■ value <p>For more information about these properties, see Setting Object Members.</p>	2018.2
options.title	string	Optional	The name for a saved search. The title property is required to save a search with Search.save() .	2015.2
options.id	string	Optional	Script ID for a saved search. If you do not set the saved search ID, NetSuite generates one for you. See Search.id .	2015.2
options.isPublic	boolean true false	Optional	<p>Set to true to make the search public. Otherwise, set to false. If you do not set this parameter, it defaults to false.</p> <p>This parameter sets the value for the Search.isPublic property.</p>	2016.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	Required parameter is missing.
SSS_INVALID_SRCH_FILTER_EXPR	The options.filters parameter is not a valid search filter, filter array, or filter expression.
SSS_INVALID_SRCH_COL	The options.columns parameter is not a valid column, string, or column or string array.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySalesOrderSearch = search.create({
4   type: search.Type.SALES_ORDER,
5   title: 'My Second SalesOrder Search',
6   id: 'customsearch_my_second_so_search',
7   columns: [
8     { name: 'entity' },
9     { name: 'subsidiary' },
10    { name: 'name' },
11    { name: 'currency' }
12  ],
13  filters: [
14    { name: 'mainline', operator: 'is', values: ['T'] }
15  ],
16  settings: [
17    { name: 'consolidationtype', value: 'AVERAGE' }
18  ]
19 });
20 ...
21 //Add additional code
22
23
24
25
26
27

```

search.create.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new search asynchronously and returns it as a search.Search object.
	<p>Note: The parameters and errors thrown for this method are the same as those for search.create(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	search.create(options)
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/search Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 search.create.promise({
4     type: search.Type.SALES_ORDER
5 })
6     .then(function(result) {
7         log.debug({
8             details: "Completed: " + result
9         });
10        // do something after completion
11    })
12    .catch(function(reason) {
13        log.debug({
14            details: "Failed: " + reason
15        });
16        // do something on failure
17    });
18 ...
19 //Add additional code

```

search.createSetting(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a new search setting and returns it as a search.setting object.</p> <p>Search settings let you specify search parameters that are typically available only in the UI. The following settings are supported:</p> <ul style="list-style-type: none"> ■ Consolidated Exchange Rate: This setting affects how consolidation is performed (for example, consolidation using the Average rate type, consolidation using the Historical rate type, and so on). This setting applies to transaction searches, and it is applicable only to OneWorld accounts. ■ Show Period End Transactions: This setting indicates whether period end transactions are included in search results. This setting applies to transaction searches, and it is applicable only to OneWorld accounts. It also requires the Period End Journal Entries feature to be enabled. <p>After you create your settings, assign them as array values to Search.settings.</p>
Returns	search.setting
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/search Module
Since	2018.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.name	string	Required	<p>The name of the search parameter to set. This value sets the Setting.name property.</p> <p>Use one of the following values for this parameter:</p> <ul style="list-style-type: none"> ■ consolidationtype: This value corresponds to the Consolidated Exchange Rate setting. ■ includeperiodendtransactions: This value corresponds to the Show Period End Transactions setting. 	2018.2
options.value	string	Required	<p>The value of the search parameter. If you are executing a joined search, this value is the join ID used for the search field specified by the options.name parameter. This value sets the Setting.value property.</p> <p>If you specify consolidationtype as the search parameter name, use one of the following values for this parameter:</p> <ul style="list-style-type: none"> ■ ACCTTYPE ■ AVERAGE ■ CURRENT ■ HISTORICAL ■ NONE <p>The default value is ACCTTYPE, which represents the type of consolidation associated with the account.</p> <p>If you specify includeperiodendtransactions as the search parameter name, use one of the following values for this parameter:</p> <ul style="list-style-type: none"> ■ F ■ FALSE ■ T ■ TRUE <p>The default value is false.</p> <p>These values are not case sensitive.</p>	2018.2

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.
SSS_INVALID_SRCH_SETTING	An unknown search parameter name is provided.
SSS_INVALID_SRCH_SETTING_VALUE	An unsupported value is set for the provided search parameter name.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.create({
4     type: 'transaction',
5     columns: [ 'trandate', 'amount', 'entity' ],
6     filters: [
7         search.createFilter({
8             name: 'internalid',
9             operator: search.Operator.ANYOF,
10            values: [13, 12356]
11        }),
12    settings: [
13        search.createSetting({
14            name: 'consolidationtype',
15            value: 'NONE'
16        })
17    ]
18 });
19 ...
20 //Add additional code

```

search.load(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Loads an existing saved search and returns it as a search.Search . The saved search could have been created using the UI or created with search.create(options) and Search.save() . The <code>search.load(options)</code> method also includes a promise version, <code>search.load.promise(options)</code> . For more information about promises, see Promise Object .
Returns	search.Search
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	5 units
Module	N/search Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.id</code>	string	Required	Internal ID or script ID of a saved search. The script ID starts with <code>customsearch</code> . See Search.id .	2015.2
<code>options.type</code>	string	Required if the saved search to load uses a standalone	The search type of the saved search to load. Use a value from the search.Type enum for this parameter.	2015.2

Parameter	Type	Required / Optional	Description	Since
		search type, optional otherwise	<p>This parameter is required if the saved search to load uses a standalone search type. A standalone search type is a search type that does not have a corresponding record type. Typically, the search type of the saved search can be determined automatically based on the corresponding record type. In this case, this parameter is not required. For standalone search types, you must specify the search type explicitly using this parameter.</p> <p>The following is a list of standalone search types:</p> <ul style="list-style-type: none"> ■ DeletedRecord ■ EndToEndTime ■ ExpenseAmortPlanAndSchedule ■ RevRecPlanAndSchedule ■ GLinesAuditLog ■ Crosschargeable ■ FinRptAggregateFR ■ BillingAccountBillCycle ■ BillingAccountBillRequest ■ BinItemBalance ■ PaymentEvent ■ Permission ■ GatewayNotification ■ TimeApproval ■ RecentRecord ■ Role ■ SavedSearch ■ ShoppingCart ■ SubscriptionRenewalHistory ■ SuiteScriptDetail ■ SupplyChainSnapshotDetails ■ SystemNote ■ TaxDetail ■ TimesheetApproval ■ Uber ■ ResAllocationTimeOffConflict ■ ComSearchOneWaySyn ■ ComSearchGroupSyn ■ Installment ■ InventoryBalance ■ InventoryNumberBin ■ InventoryNumberItem ■ InventoryStatusLocation ■ InvtNumberItemBalance ■ ItemBinNumber 	

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	{1}: Missing a required argument: {2}	Required parameter is missing.
INVALID_SEARCH	That search or mass update does not exist.	Cannot find saved search with the saved search ID from options.id parameter.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.load({
4   id: 'customsearch_my_so_search'
5 });
6 ...
7 //Add additional code

```

search.load.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Loads an existing saved search asynchronously and returns it as a search.Search object. The saved search could have been created using the UI or created with search.create(options) and Search.save() . i Note: The parameters and errors thrown for this method are the same as those for search.load(options). For more information on promises, see Promise Object.
Returns	Promise Object
Synchronous Version	search.load(options)
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	5 units
Module	N/search Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```
1 //Add additional code
```

```

2 ...
3 search.load.promise({
4     type : search.Type.SALES_ORDER,
5     id : 'customsearch_txn_search_salesorder'
6   })
7   .then(function (result) {
8     log.debug({
9       details: "Completed: " + result
10    });
11    // do something after completion
12  })
13  .catch(function onRejected(reason) {
14    // do something on rejection
15  });
16 ...
17 //Add additional code

```

search.delete(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Deletes an existing saved search. The saved search could have been created using the UI or created with search.create(options) and Search.save() . The search.delete(options) method also includes a promise version, search.delete.promise(options). For more information about promises, see Promise Object .
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	5 units
Module	N/search Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	Required	Internal ID or script ID of a saved search. The script ID starts with customsearch. See Search.id .	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	{1}: Missing a required argument: {2}	Required parameter is missing.
INVALID_SEARCH	That search or mass update does not exist.	Cannot find saved search with the saved search ID from options.id parameter.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 search.delete({
4     id: 'customsearch_my_so_search'
5 });
6 ...
7 //Add additional code

```

search.delete.promise(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Deletes an existing saved search asynchronously and returns it as a search.Search object. The saved search can be created using the UI or created with search.create(options) and Search.save() .
	Note: The parameters and errors thrown for this method are the same as those for search.delete(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	search.delete(options)
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	5 units
Module	N/search Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 search.delete.promise({id: 'customsearch_txn_search_salesorder'})
4     .then(function(){
5         search.load({
6             id: 'customsearch_txn_search_salesorder'
7         });
8     })
9     .catch(function onRejected(reason) {
10         log.debug({
11             details: 'Invalid search: ' + reason.name
12         });
13     });
14 ...
15 //Add additional code

```

search.duplicates(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Performs a search for duplicate records based on the account's duplicate detection configuration.</p> <p>This method also includes a promise version, search.duplicates.promise(options). For more information about promises, see Promise Object.</p> <p> Important: This API works only for records that support duplicate record detection (for example, customer, lead, prospect, contact, partner, and vendor records).</p> <p>For more information about duplicate record detection, see the help topic Duplicate Record Detection.</p>
Returns	<p>search.Result[] that contains the duplicate records</p> <p>Results are limited to 1000 rows.</p> <p>If there are no search results, this method returns null.</p>
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	10 units
Module	N/search Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	enum	Required	<p>The search type that you want to check for duplicates.</p> <p>Use the search.Type enum for this parameter. The type you specify must correspond to a record type that supports duplicate record detection (for example, customer, lead, prospect, contact, partner, and vendor records).</p>	2015.2
options.fields	Object	Optional	<p>A set of key/value pairs used to detect duplicates. The keys are internal ID names of the fields used to detect duplicates. You can specify fields such as companyname, email, name, phone, address1, city, state, and zipcode. For example, to detect duplicates based on the value of the email field, use 'email' : 'sample@test.com'.</p> <p>Use this parameter to specify the fields (and their values) to use to detect duplicates. If you are searching for duplicates based on fields that appear on a certain record type, this parameter is required. If you are searching for duplicates of a specific record (of the specified type), use the options.id parameter instead.</p>	2015.2

Parameter	Type	Required / Optional	Description	Since
options.id	number	Optional	<p>Internal ID of an existing record.</p> <p>Use this parameter to specify a record to detect duplicates of. The duplicate record detection settings in the account determine which fields are used to detect duplicates of the specified record. If you are searching for duplicates of a specific record (of the specified type), this parameter is required. If you are searching for duplicates based on fields that appear on a certain record type, use the options.fields parameter instead.</p>	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	{1}: Missing a required argument: {2}	Required parameter is missing.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 // Search for duplicates of a specific record using the options.id
4 // parameter
5 var duplicatesOfRecord = search.duplicates({
6   type: search.Type.CONTACT,
7   id: 425
8 });
9
10 // Search for duplicates based on specific fields on a record type
11 // using the options.fields parameter
12 var duplicatesUsingFields = search.duplicates({
13   type: search.Type.CONTACT,
14   fields: {
15     'email' : 'sample@test.com'
16   }
17 });
18 ...
19 //Add additional code

```

search.duplicates.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs a search for duplicate records asynchronously based on the Duplicate Detection configuration for the account. Returns an array of search.Result objects. This method only applies to records that support duplicate record detection. These records include customer lead prospect partner vendor contact.
 Note: The parameters and errors thrown for this method are the same as those for search.duplicates(options) . For more information on promises, see Promise Object .	
Returns	Promise Object

Synchronous Version	search.duplicates(options)
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units
Module	N/search Module
Since	2015.2

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 search.duplicates.promise({
4     type: search.Type.CUSTOMER,
5     id: 28
6     })
7 .then(function (result) {
8     log.debug({
9         details: "Completed: " + result
10    });
11    // do something after completion
12   })
13 .catch(function onRejected(reason) {
14     // do something on rejection
15   });
16 ...
17 //Add additional code

```

search.global(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs a global search against a single keyword or multiple keywords. Similar to the global search functionality in the UI, you can programmatically filter the global search results that are returned. For example, you can use the following filter to limit the returned records to Customer records: 'cu: simpson' The search.global(options) method also includes a promise version, search.global.promise(options). For more information about promises, see Promise Object . For more information about global search, see the help topic Global Search .
Returns	<code>search.Result[]</code> as an array of result objects containing these columns: name, type, info1, and info2 Results are limited to 1000 records. If there are no search results, this method returns null.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	10 units
Module	N/search Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.keywords	string	Required	Global search keywords string or expression.	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	{1}: Missing a required argument: {2}	Required parameter is missing.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var customerSearch = search.global({
4   keywords: 'cu: simpson'
5 });
6 ...
7 //Add additional code

```

search.global.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs a global search asynchronously against a single keyword or multiple keywords. Returns an array of search.Result objects with four columns: name , type , info1 , and info2 .  Note: The parameters and errors thrown for this method are the same as those for search.global(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	search.global(options)
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	10 units

Module	N/search Module
Since	2015.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 search.global.promise({
4     keywords: 'Alan Rath'
5 })
6 .then(function (result) {
7     log.debug({
8         details: "Completed: " + result
9     });
10    // do something after completion
11 })
12 .catch(function onRejected(reason) {
13     // do something on rejection
14 });
15 ...
16 //Add additional code

```

search.lookupFields(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Performs a search for one or more body fields on a record.</p> <p>You can use joined-field lookups with this method, with the following syntax:</p> <p><code>join_id.field_name</code></p> <p>The <code>search.lookupFields(options)</code> method also includes a promise version, <code>search.lookupFields.promise(options)</code>. For more information about promises, see Promise Object.</p> <p>Note that the return contains either an object or a scalar value, depending on whether the looked-up field holds a single value, or a collection of values. Single select fields are returned as an object with value and text properties. Multi-select fields are returned as an object with value:text pairs.</p> <p>In the following example, a select field like <code>my_select</code> would return an array of objects containing a value and text property. This select field contains multiple entries to select from, so each entry would have a numerical id (the value) and a text display (the text).</p>
---------------------------	---

	<p>For "internalid" in this particular code snippet, the sample returns 1234. The internal id of a record is a single value, so a scalar is returned.</p> <pre> 1 { 2 internalid: 1234, 3 firstname: 'Joe', 4 my_select: [5 { 6 value: 1, 7 text: 'US Sub' 8 }], 9 my_multiselect: [10 { 11 value: 1, 12 text: 'US Sub' 13 }, 14 { 15 value: 2, 16 text: 'EU Sub' 17 } 18 } </pre> <p>If you try to look up a field that does not exist on the specified record, an SSS_INVALID_SRCH_COL error is thrown.</p>
Returns	Object Object[] <ul style="list-style-type: none"> ▪ Returns select fields as an object with value and text properties. ▪ Returns multiselect fields as an array of object with value:text pairs.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	1 unit
Module	N/search Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	enum	Required	The search type for which you want to look up fields. Use the search.Type enum for this argument.	2015.2
options.id	string	Required	Internal ID for the record, for example 777 or 87.	2015.2
options.columns	string string[]	Required	Array of column/field names to look up, or a single column/field name. The columns parameter can also be set to reference joined fields.	2015.2

Errors

Error Code	Message	Thrown If
SSS_INVALID_SRCH_COL	An nlobjSearchColumn contains an invalid column, or is not in proper syntax: {1}.	The options.columns parameter includes invalid columns for the specified record.
SSS_MISSING_REQD_ARGUMENT	{1}: Missing a required argument: {2}	Required parameter is missing.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var fieldLookUp = search.lookupFields({
4   type: search.Type.SALES_ORDER,
5   id: '87',
6   columns: ['entity', 'subsidiary', 'name', 'currency']
7 });
8 ...
9 //Add additional code

```

search.lookupFields.promise(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Performs a search asynchronously for one or more body fields on a record. Returns select fields as an object with value and text properties. Returns multiselect fields as an object with value:text pairs.
	 Note: The parameters and errors thrown for this method are the same as those for search.lookupFields(options) . For more information on promises, see Promise Object .
Returns	Promise Object
Synchronous Version	search.lookupFields(options)
Supported Script Types	All client-side scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	1 unit
Module	N/search Module
Since	2015.2

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```
1 //Add additional code
```

```

2 ...
3 search.lookupFields.promise({
4     type: search.Type.EMPLOYEE,
5     id: -5,
6     columns : 'email'
7 })
8 .then(function (result) {
9     log.debug({
10         details: "Completed: " + result
11     });
12     // do something after completion
13 })
14 .catch(function onRejected(reason) {
15     // do something on rejection
16 });
17 ...
18 //Add additional code

```

search.createColumn(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new search column as a search.Column object.
	<p>Important: As you create search columns, consider the following:</p> <ul style="list-style-type: none"> ■ You cannot directly create a filter or column for a list/record type field in SuiteScript by passing in its text value. You must use the field's internal ID. If you must use the field's text value, you can create a filter or column with a formula using name: 'formulatext'. ■ After you create a column, you cannot change the sort order of the column. If you use the same column in another search and specify a new sort order, the previous sort order is still used (the sort order that you specified using the options.sort parameter).
Returns	search.Column
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/search Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.name	string	Required	Name of the search column. See Column.name .	2015.2

Parameter	Type	Required / Optional	Description	Since
options.join	string	Optional	Join ID for the search column. See Column.join .	2015.2
options.summary	string	Optional	Summary type for the column. See search.Summary and Column.summary .	2015.2
options.formula	string	Optional	Formula for the search column. See Column.formula .	2015.2
options.function	string	Optional	Special function for the search column. See Column.function .	2015.2
options.label	string	Optional	Label for the search column. See Column.label .	2015.2
options.sort	string	Optional	The sort order of the column. Use the search.Sort enum for this argument. Also see Column.sort .	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	{1}: Missing a required argument: {2}	Required parameter is missing.
SSS_INVALID_SRCH_COLUMN_SUM	A search.Column object contains an invalid column summary type, or is not in proper syntax: {1}.	The options.summary parameter is not a valid search summary type. See Search.Summary .
INVALID_SRCH_FUNCTN		An unknown function is provided.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var currencyColumn = search.createColumn({
4   name: 'currency',
5   sort: search.Sort.ASC
6 });
7 ...
8 //Add additional code

```

search.createFilter(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description

Creates a new search filter as a [search.Filter](#) object.

	 Important: You cannot directly create a filter or column for a list/record type field in SuiteScript by passing in its text value. You must use the field's internal ID. If you must use the field's text value, you can create a filter or column with a formula using name: 'formulatext'.
Returns	search.Filter
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/search Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.name	string	Required	Name or internal ID of the search field.	2015.2
options.join	string	Optional	Join ID for the search filter.	2015.2
options.operator	string	Required	Operator used for the search filter. Use the search.Operator enum.	2015.2
options.values	string Date number boolean string[] Date[] number[]	Optional	Values to be used as filter parameters.	2015.2
options.formula	string	Optional	Formula used by the search filter.	2015.2
options.summary	string	Optional	Summary type for the search filter. See search.Summary .	2015.2

Errors

Error Code	Message	Thrown If
SSS_MISSING_REQD_ARGUMENT	{1}: Missing a required argument: {2}	Required parameter is missing.
SSS_INVALID_SRCH_SUMMARY_TYP	A search.Column object contains an invalid column summary type, or is not in proper syntax: {1}.	options.summary parameter is not a valid search summary type. See search.Summary .
SSS_INVALID_SRCH_OPERATOR	An search.Filter object contains an invalid operator, or is not in proper syntax: {1}.	options.operator parameter is not a valid operator type. See search.Operator .

Syntax

The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 // Add additional code
2 ...
3 // Create a filter joined to another record type. When you create a joined filter:
4 // - The name property is the field ID of the field in the joined record that you are filtering on
5 // - The join property is the field ID of the field in the current record that contains the record
6 // type you want to join to
7 // - The operator property is the operator to use to filter the results
8 // - The values property contains the values to use to filter the results
9 //
10 // For example, the following search definition lists the first 100 employees found
11 // who have a custom role. The search definition specifies that the search applies to
12 // Employee records. The filter definition joins the Role record type to the search
13 // and returns results where the iscustom field (a field on the Role record) is true.
14 var result = search.create({
15   type: 'employee',
16   columns: ['firstname', 'lastname', 'role'],
17   filters: [
18     search.createFilter({
19       name: 'iscustom',
20       join: 'role',
21       operator: search.Operator.IS,
22       values: true
23     })
24   ]
25 }).run().getRange({
26   start: 0,
27   end: 100
28 });
29 log.debug({
30   title: 'Result',
31   details: result
32 });
33 ...
34 // Add additional code

```

search.Operator



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Enumeration that holds the values for search operators to use with the search.Filter.</p> <p>See the help topic SuiteScript 1.0 Search Operators for more information about the field types supported for each operator type.</p>
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	<p>All script types</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/search Module
Since	2015.2

Values

■ AFTER	■ IS	■ NOTGREATERTHANOREQUALTO
---------	------	---------------------------

■ ALLOF	■ ISEMPY	■ NOTLESSTHAN
■ ANY	■ ISNOT	■ NOTLESSTHANOREQUALTO
■ ANYOF	■ ISNOTEMPTY	■ NOTON
■ BEFORE	■ LESSTHAN	■ NOTONRAFTER
■ BETWEEN	■ LESSTHANOREQUALTO	■ NOTONRBEFORE
■ CONTAINS	■ NONEOF	■ NOTWITHIN
■ DOESNOTCONTAIN	■ NOTAFTER	■ ON
■ DOESNOTSTARTWITH	■ NOTALLOF	■ ONRAFTER
■ EQUALTO	■ NOTBEFORE	■ ONRBEFORE
■ GREATERTHAN	■ NOTBETWEEN	■ STARTSWITH
■ GREATERTHANOREQUALTO	■ NOTEQUALTO	■ WITHIN
■ HASKEYWORDS	■ NOTGREATERTHAN	

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code...
2 var mySearchFilter = search.createFilter({
3   name: 'entity',
4   operator: search.Operator.ISEMPY
5 });
6 ...
7 //Add additional code
8

```

search.Sort

Enum Description	Enumeration that holds the values for supported sorting directions used with search.createColumn(options) .
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Values

■ ASC
■ DESC
■ NONE

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var currencyColumn = search.createColumn({
4   name: 'currency',
5   sort: search.Sort.ASC
6 });
7 ...
8 //Add additional code

```

search.Summary



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Enumeration that holds the values for summary types used by the Column.summary or Filter.summary properties. For more information about each summary type, see the help topic SuiteScript 1.0 Search Summary Types.</p> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Values

<ul style="list-style-type: none"> ▪ GROUP ▪ COUNT ▪ SUM ▪ AVG ▪ MIN ▪ MAX
--

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearchFilter = search.createFilter({
4   name: 'entity',
5   summary: search.Summary.GROUP
6 });

```

```

7 | ...
8 | //Add additional code

```

search.Type



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Enumeration that holds the string values for search types supported in the N/search Module . This enum is used to pass the type argument to search.create(options) .
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/search Module
Since	2015.2

Values



Note: A search type is not a synonym for a record type. The supported search types listed below do not necessarily correspond with the supported record types listed in the [N/record Module](#).

<ul style="list-style-type: none"> ■ ACCOUNT ■ ACCOUNTING_BOOK ■ ACCOUNTING_CONTEXT ■ ACCOUNTING_PERIOD ■ ACTIVITY ■ ADV_INTER_COMPANY_JOURNAL_ENTRY ■ AGGR_FIN_DAT ■ ALLOC_RECOMMENDATION_DEMAND ■ ALLOC_RECOMMENDATION_DETAIL ■ AMORTIZATION_SCHEDULE ■ AMORTIZATION_TEMPLATE ■ ASSEMBLY_BUILD ■ ASSEMBLY_ITEM ■ ASSEMBLY_UNBUILD ■ AUTHENTICATE_DEVICE_INPUT ■ BALANCE_TRX_BY_SEGMENTS ■ BALANCING_DETAIL ■ BALANCING_RESULT ■ BALANCING_TRANSACTION ■ BILLING_ACCOUNT ■ BILLING_ACCOUNT_BILL_CYCLE ■ BILLING_ACCOUNT_BILL_REQUEST 	<ul style="list-style-type: none"> ■ FIN_RPT_AGGREGATE_F_R ■ FINANCIAL_INSTITUTION ■ FIXED_AMOUNT_PROJECT_REVENUE_RULE ■ FOLDER ■ FULFILLMENT_REQUEST ■ GATEWAY_NOTIFICATION ■ GENERIC_RESOURCE ■ GIFT_CERTIFICATE ■ GIFT_CERTIFICATE_ITEM ■ GLOBAL_ACCOUNT_MAPPING ■ GLOBAL_INVENTORY_RELATIONSHIP ■ GL_LINES_AUDIT_LOG ■ GL_NUMBERING_SEQUENCE ■ GOAL ■ INBOUND_SHIPMENT ■ INSTALLMENT ■ INTER_COMPANY_JOURNAL_ENTRY ■ INTER_COMPANY_TRANSFER_ORDER ■ INVENTORY_ADJUSTMENT ■ INVENTORY_BALANCE ■ INVENTORY_COST_REVALUATION ■ INVENTORY_COUNT 	<ul style="list-style-type: none"> ■ PROJECT_TEMPLATE ■ PROMOTION_CODE ■ PROSPECT ■ PURCHASE_CONTRACT ■ PURCHASE_ORDER ■ PURCHASE_REQUSITION ■ RECENT_RECORD ■ RES_ALLOCATION_TIME_OFF_CONFLICT ■ RESOURCE_ALLOCATION ■ RESTLET ■ RETURN_AUTHORIZATION ■ REVENUE_ARRANGEMENT ■ REVENUE_COMMITMENT ■ REVENUE_COMMITMENT_REVERSAL ■ REVENUE_PLAN ■ REV_REC_PLAN_AND_SCHEDULE ■ REV_REC_SCHEDULE ■ REV_REC_TEMPLATE ■ ROLE ■ S_C_M_PREDICTED_RISKS
---	---	---

<ul style="list-style-type: none"> ■ BILLING_CLASS ■ BILLING_RATE_CARD ■ BILLING_REVENUE_EVENT ■ BILLING_SCHEDULE ■ BIN ■ BIN_ITEM_BALANCE ■ BIN_TRANSFER ■ BIN_WORKSHEET ■ BLANKET_PURCHASE_ORDER ■ BOM ■ BOM_REVISION ■ BONUS ■ BONUS_TYPE ■ BUDGET_EXCHANGE_RATE ■ BULK_OWNERSHIP_TRANSFER ■ BUNDLE_INSTALLATION_SCRIPT ■ CALENDAR_EVENT ■ CAMPAIGN ■ CARDHOLDER_AUTHENTICATION ■ CARDHOLDER_AUTHENTICATION_EVENT ■ CASH_REFUND ■ CASH_SALE ■ CHALLENGE_SHOPPER_INPUT ■ CHARGE ■ CHARGE_RULE ■ CHECK ■ CLASSIFICATION ■ CLIENT_SCRIPT ■ CMS_CONTENT ■ CMS_CONTENT_TYPE ■ CMS_PAGE ■ COM_SEARCH_BOOST ■ COM_SEARCH_BOOST_TYPE ■ COM_SEARCH_GROUP_SYN ■ COM_SEARCH_ONE_WAY_SYN ■ COMMERCE_CATEGORY ■ COMMERCE_SEARCH_ACTIVITY_DATA ■ COMPETITOR ■ CONSOLIDATED_EXCHANGE_RATE ■ CONTACT ■ CONTACT_CATEGORY ■ CONTACT_ROLE ■ COST_CATEGORY ■ COUPON_CODE ■ CREDIT_CARD_CHARGE ■ CREDIT_CARD_REFUND ■ CREDIT_MEMO 	<ul style="list-style-type: none"> ■ INVENTORY_DEMAND ■ INVENTORY_DETAIL ■ INVENTORY_ITEM ■ INVENTORY_NUMBER ■ INVENTORY_NUMBER_BIN ■ INVENTORY_NUMBER_ITEM ■ INVENTORY_STATUS ■ INVENTORY_STATUS_CHANGE ■ INVENTORY_STATUS_LOCATION ■ INVENTORY_TRANSFER ■ INVOICE ■ INVOICE_GROUP ■ INVNT_NUMBER_ITEM_BALANCE ■ ISSUE ■ ITEM ■ ITEM_ACCOUNT_MAPPING ■ ITEM_BIN_NUMBER ■ ITEM_COLLECTION ■ ITEM_COLLECTION_ITEM_MAP ■ ITEM_DEMAND_PLAN ■ ITEM_FULFILLMENT ■ ITEM_GROUP ■ ITEM_LOCATION_CONFIGURATION ■ ITEM_RECEIPT ■ ITEM_REVISION ■ ITEM_SUPPLY_PLAN ■ JOB ■ JOB_STATUS ■ JOB_TYPE ■ JOURNAL_ENTRY ■ KIT_ITEM ■ LABOR_BASED_PROJECT_REVENUE_RULE ■ LEAD ■ LOCATION ■ LOT_NUMBERED_ASSEMBLY_ITEM ■ LOT_NUMBERED_INVENTORY_ITEM ■ MANUFACTURING_COST_TEMPLATE ■ MANUFACTURING_OPERATION_TASK ■ MANUFACTURING_ROUTING ■ MAP_REDUCE_SCRIPT ■ MARKUP_ITEM ■ MASSUPDATE_SCRIPT ■ MEM_DOC ■ MERCHANDISE_HIERARCHY_LEVEL ■ MERCHANDISE_HIERARCHY_NODE ■ MERCHANDISE_HIERARCHY_VERSION 	<ul style="list-style-type: none"> ■ S_C_M_PREDICTION_TRAIN_HISTORY ■ SALES_ORDER ■ SALES_ROLE ■ SALES_TAX_ITEM ■ SAVED_SEARCH ■ SCHEDULED_SCRIPT ■ SCHEDULED_SCRIPT_INSTANCE ■ SCRIPT_DEPLOYMENT ■ SERIALIZED_ASSEMBLY_ITEM ■ SERIALIZED_INVENTORY_ITEM ■ SERVICE_ITEM ■ SHIP_ITEM ■ SHOPPING_CART ■ SOLUTION ■ STATE ■ STATISTICAL_JOURNAL_ENTRY ■ STORE_PICKUP_FULFILLMENT ■ SUBSCRIPTION ■ SUBSCRIPTION_CHANGE_ORDER ■ SUBSCRIPTION_LINE ■ SUBSCRIPTION_LINE_REVISION ■ SUBSCRIPTION_PLAN ■ SUBSCRIPTION_RENEWAL_HISTORY ■ SUBSIDIARY ■ SUBTOTAL_ITEM ■ SUITELET ■ SUITE_SCRIPT_DETAIL ■ SUPPLY_CHAIN_SNAPSHOT ■ SUPPLY_CHAIN_SNAPSHOT_DETAILS ■ SUPPORT_CASE ■ SYSTEM_NOTE ■ TASK ■ TAX_DETAIL ■ TAX_GROUP ■ TAX_PERIOD ■ TAX_TYPE ■ TERM ■ TIMESHEET_APPROVAL ■ TIME_APPROVAL ■ TIME_BILL ■ TIME_ENTRY ■ TIME_OFF_CHANGE ■ TIME_OFF_PLAN ■ TIME_OFF_REQUEST ■ TIME_OFF_RULE ■ TIME_OFF_TYPE
--	--	---

<ul style="list-style-type: none"> ■ CROSSCHARGEABLE ■ CURRENCY ■ CURRENCY_EXCHANGE_RATE ■ CUSTOMER ■ CUSTOMER_CATEGORY ■ CUSTOMER_DEPOSIT ■ CUSTOMER_MESSAGE ■ CUSTOMER_PAYMENT ■ CUSTOMER_PAYMENT_Authorization ■ CUSTOMER_REFUND ■ CUSTOMER_STATUS ■ CUSTOMER_SUBSIDIARY_RELATIONSHIP ■ CUSTOM_RECORD ■ CUSTOM_TRANSACTION ■ DELETED_RECORD ■ DEPARTMENT ■ DEPOSIT ■ DEPOSIT_APPLICATION ■ DESCRIPTION_ITEM ■ DISCOUNT_ITEM ■ DOWNLOAD_ITEM ■ EMPLOYEE ■ EMPLOYEE_CHANGE_REQUEST ■ EMPLOYEE_CHANGE_REQUEST_TYPE ■ EMPLOYEE_CHANGE_TYPE ■ EMPLOYEE_PAYROLL_ITEM ■ EMPLOYEE_STATUS ■ END_TO_END_TIME ■ ENTITY ■ ENTITY_ACCOUNT_MAPPING ■ ESTIMATE ■ EXPENSE_AMORTIZATION_EVENT ■ EXPENSE_AMORT_PLAN_AND_SCHEDULE ■ EXPENSE_CATEGORY ■ EXPENSE_PLAN ■ EXPENSE_REPORT ■ FAIR_VALUE_PRICE 	<ul style="list-style-type: none"> ■ MESSAGE ■ MFG_PLANNED_TIME ■ NEXUS ■ NON_INVENTORY_ITEM ■ NOTE ■ NOTE_TYPE ■ OPPORTUNITY ■ OTHER_CHARGE_ITEM ■ OTHER_NAME ■ OTHER_NAME_CATEGORY ■ PARTNER ■ PARTNER_CATEGORY ■ PAYCHECK ■ PAYCHECK_JOURNAL ■ PAYMENT_EVENT ■ PAYMENT_INSTRUMENT ■ PAYMENT_ITEM ■ PAYMENT_METHOD ■ PAYMENT_OPTION ■ PAYMENT_RESULT_PREVIEW ■ PAYROLL_SETUP ■ PAYROLL_ITEM ■ PCT_COMPLETE_PROJECT_REVENUE_RULE ■ PERFORMANCE_METRIC ■ PERFORMANCE REVIEW ■ PERFORMANCE REVIEW SCHEDULE ■ PERIOD_END_JOURNAL ■ PERMISSION ■ PHONE_CALL ■ PORTLET ■ PRICE_BOOK ■ PRICE_LEVEL ■ PRICE_PLAN ■ PRICING ■ PRICING_GROUP ■ PROJECT_EXPENSE_TYPE ■ PROJECT_IC_CHARGE_REQUEST ■ PROJECT_TASK 	<ul style="list-style-type: none"> ■ TIME_SHEET ■ TOPIC ■ TRANSACTION ■ TRANSFER_ORDER ■ UBER ■ UNITS_TYPE ■ USAGE ■ USEREVENT_SCRIPT ■ VENDOR ■ VENDOR_BILL ■ VENDOR_CATEGORY ■ VENDOR_CREDIT ■ VENDOR_PAYMENT ■ VENDOR_PREPAYMENT_APPLICATION ■ VENDOR_RETURN_Authorization ■ VENDOR_SUBSIDIARY_RELATIONSHIP ■ WAVE ■ WBS ■ WEBSITE ■ WORKFLOW_ACTION_SCRIPT ■ WORK_ORDER ■ WORK_ORDER_CLOSE ■ WORK_ORDER_COMPLETION ■ WORK_ORDER_ISSUE ■ WORKPLACE ■ ZONE
--	--	--

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/search Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var mySearch = search.create({
4   type: search.Type.CUSTOMER,

```

```

5   filters: filters,
6   columns: columns
7 });
8 ...
9 //Add additional code

```

N/sftp Module

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

The SFTP module provides a way to manage folders and upload or download files from external SFTP servers.

SFTP servers can be hosted by your organization or by a third party. NetSuite does not provide SFTP server functionality. All SFTP transfers to or from NetSuite must originate from SuiteScript. It is not possible for external clients to initiate file transfers using SFTP.

Use SSH keys to establish an SFTP connection. By using the keys, you can manage files and directories by using the SSH file transfer (SFTP) protocol. For more information, see the help topic [SSH Keys for SFTP](#). For more information about the N/keyControl Module, see [N/keyControl Module](#).

 **Important:** All paths, directories, and filenames that contain wildcards such as ? and * must have those characters escaped, unless these characters are specifically intended to work as wildcards.

 **Note:** To use an external server to initiate a NetSuite file transfer that doesn't use SFTP, you can use RESTlets or SOAP web services. In SuiteScript, RESTlets can respond to requests containing file data and save them in the File Cabinet. RESTlets can also respond to requests for file data by loading the contents from the File Cabinet and returning them in the response. Note that binary file content must be received or sent as Base64 encoded Strings. See the help topic [SuiteScript 2.0 RESTlet Script Type](#) for more information.

In SOAP web services, applications can invoke CRUD operations on the file record to populate or change the contents of the File Cabinet. See the help topics [SuiteTalk SOAP Web Services Platform Guide](#) and [File](#) for more information.

- [N/sftp Module Members](#)
- [Connection Object Members](#)
- [N/sftp Module Script Samples](#)
- [Setting up an SFTP Transfer](#)
- [SFTP Authentication](#)
- [Supported Cipher Suites and Host Key Types](#)
- [Supported SuiteScript File Types](#)
- [N/keyControl Module](#)

N/sftp Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	sftp.Connection	Object	Server-side scripts	Represents a connection to the account on the remote FTP server.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	sftp.createConnection(options)	sftp.Connection	Server-side scripts	Establishes a connection to a remote FTP server.
Enum	sftp.MAX_CONNECT_TIMEOUT	Enum	Server-side scripts	Holds the values for maximum connection timeout.
	sftp.MIN_CONNECT_TIMEOUT	Enum	Server-side scripts	Holds the values for minimum connection timeout.
	sftp.MAX_PORT_NUMBER	Enum	Server-side scripts	Holds the values for the maximum port number.
	sftp.MIN_PORT_NUMBER	Enum	Server-side scripts	Holds the values for the minimum port number.
	sftp.DEFAULT_PORT_NUMBER	Enum	Server-side scripts	Holds the values for the default port number.
	sftp.Sort	Enum	Server-side scripts	Holds the values to be used to sort listed directory.

Connection Object Members

The following members are called on the [sftp.Connection](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Connection.download(options)	file.File	Server-side scripts	Downloads a file from the remote FTP server.
	Connection.upload(options)	void	Server-side scripts	Uploads a file to the remote FTP server.
	Connection.makeDirectory(options)	string	Server-side scripts	Creates an empty directory.
	Connection.removeDirectory(options)	void	Server-side scripts	Removes an empty directory.
	Connection.removeFile(options)	void	Server-side scripts	Removes a file in a directory.
	Connection.move(options)	void	Server-side scripts	Moves a file or directory from one location to another.
	Connection.list(options)	Array<Object>	Server-side scripts	Lists the remote directory.
Enum	Connection.MAX_FILE_SIZE	number	Server-side scripts	Holds the values for the maximum file size.
	Connection.MAX_TRANSFER_TIMEOUT	number	Server-side scripts	Holds the values for the maximum transfer timeout.

N/sftp Module Script Samples

i Note: These sample scripts use the require function so that you can copy it into the debugger and test them. Keep in mind that you must use the define function in your entry point script (the script you attach to a script record). For additional information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

For help with writing scripts in SuiteScript 2.0, see the help topics [SuiteScript 2.0 Hello World](#) and [SuiteScript 2.0 Entry Point Script Creation and Deployment](#).

! **Important:** Before you run the script, you must replace the GUID and host key with one specific to your account. The user name, URL, and directory values in this sample are also placeholders. Before using the sample, replace the placeholder values with valid values from your NetSuite account.

Example 1

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to upload and download a file.

To obtain a real host key, use ssh-keyscan <domain>.

To create a real password GUID, obtain a password value from a credential field on a form. For more information, see [Form.addCredentialField\(options\)](#). Also see [N/https Module Script Sample](#) for a Suitelet sample that shows creating a form field that generates a GUID.

```

1  /**
2  * @NApiVersion 2.x
3  */
4
5  require(['N/sftp', 'N/file'],
6    function(sftp, file) {
7      var myPwdGuid = "B34672495064525E5D65032D63B52301";
8      var myHostKey = "AAA1234567890Q=";
9
10     // establish connection to remote FTP server
11
12     var connection = sftp.createConnection({
13       username: 'myuser',
14       passwordGuid: myPwdGuid, // references var myPwdGuid
15       url: 'host.somewhere.com',
16       directory: 'myuser/wheres/my/file',
17       hostKey: myHostKey // references var myHostKey
18     });
19
20     // specify the file to upload using the N/file module
21
22     var myFileToUpload = file.create({
23       name: 'originalname.js',
24       fileType: file.Type.PLAINTEXT,
25       contents: 'I am a test file.'
26     });
27
28     // upload the file to the remote server
29
30     connection.upload({
31       directory: 'relative/path/to/remote/dir',
32       filename: 'newFileNameOnServer.js',
33       file: myFileToUpload,

```

```

34     replaceExisting: true
35   });
36
37   // download the file from the remote server
38
39   var downloadedFile = connection.download({
40     directory: 'relative/path/to/file',
41     filename: 'downloadMe.js'
42   });
43 });

```

Example 2

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how you can manage files and directories.

```

1  /**
2  * @NApiVersion 2.x
3  */
4
5 require(['N/ftp', 'N/file'], function(sftp, file){
6   // Establish connection
7   log.debug('Establishing SFTP connection...');
8   var connection = sftp.createConnection({
9     username: 'sftpuser',
10    keyId: 'custkeysftp_nft_demo_key',
11    url: 'myurl',
12    port: 22,
13    directory: 'inbound',
14    hostKey: 'myhostkey'
15  });
16  log.debug('Connection established!');
17 // -----
18 // List the directory and log number of elements there
19 var list = connection.list({path: 'yyy/test'});
20 log.debug('Items in directory "test" at the beginning: ' + list.length);
21 // -----
22 // Generate the test file
23 log.debug('Generating test file...');
24 var myFileToUpload = file.create({
25   name: 'asdf.txt',
26   fileType: file.Type.PLAINTEXT,
27   contents: 'I am a test file.'
28 });
29 log.debug('Test file generated, uploading to "test" directory...');
30 // -----
31 // Upload the test file
32 connection.upload({
33   directory: 'yyy/test',
34   filename: 'af.txt',
35   file: myFileToUpload,
36   replaceExisting: true
37 });
38 log.debug('Upload complete!');
39 // -----
40 // List the directory to see there is one more file than before
41 list = connection.list({path: 'yyy/test'});
42 log.debug('Items in directory "test" after the upload: ' + list.length);
43 // -----
44 // Create new directory
45 log.debug('Creating directory "test2"...');
46 try {
47   connection.makeDirectory({path: 'yyy/test2'});
48   log.debug('Directory created.');
49 } catch (e) {

```

```

50     log.debug('Directory not created.');
51     log.error(e.message);
52 }
53 list = connection.list({path: 'yyy/test2'});
54 log.debug('Items in directory "test2": ' + list.length);
55 //-----
56 // Move the test file there
57 log.debug('Moving the test file from "test" to "test2"...');
58 connection.move({
59   from: 'yyy/test/af.txt',
60   to: 'yyy/test2/af.txt'
61 })
62 log.debug('File moved!');
63 //-----
64 // List the original directory again to see the file is moved out
65 list = connection.list({path: 'yyy/test'});
66 log.debug('Items in directory "test" after the upload: ' + list.length);
67 //-----
68 // List the new directory for the file
69 list = connection.list({path: 'yyy/test2'});
70 log.debug('Items in directory "test2" after the upload: ' + list.length);
71 log.debug(JSON.stringify(list));
72 //-----
73 // Try to remove the directory
74 log.debug('Removing directory "test2"...');
75 try {
76   connection.removeDirectory({
77     path: 'yyy/test2'
78   });
79   log.debug('Directory removed!');
80 } catch (e) {
81   log.debug('Directory not removed!');
82   log.error(e.message);
83 }
84 //-----
85 // It's not empty so let's delete the file first
86 log.debug('Removing test file from "test2" directory...');
87 connection.removeFile({
88   path: 'yyy/test2/af.txt'
89 });
90 log.debug('Test file removed!');
91
92 list = connection.list({path: 'yyy/test2'});
93 log.debug('Items in directory "test2": ' + list.length);
94 //-----
95 // Remove it now again
96 log.debug('Removing directory "test2"...');
97 try {
98   connection.removeDirectory({
99     path: 'yyy/test2'
100   });
101   log.debug('Directory removed!');
102 } catch (e) {
103   log.debug('Directory not removed!');
104   log.error(e.message);
105 }
106 //-----
107 // Listing it again
108 log.debug('Trying to list directory "test2"...');
109 try {
110   list = connection.list({path: 'yyy/test2'});
111 } catch (e) {
112   log.error(e.message);
113 }
114 //-----
115 ;
116 })

```

Example 3

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to use the N/sftp module enums to set conditional default settings. It creates a secure connection and attempts to upload and add a large file.

```

1  /**
2   * @NApiVersion 2.x
3   * @NScriptType UserEventScript
4   * @NModuleScope SameAccount
5   */
6
7 define(['N/file', 'N/sftp', 'N/error'],
8 function(file, sftp, error) {
9     return {
10         beforeLoad: function(){
11             var portNumber = -1;
12             var connectTimeout = -1;
13             var transferTimeout = -1;
14             //these variables can be taken as parameters of the script instead
15
16             if (portNumber <sftp.MIN_PORT_NUMBER || portNumber > sftp.MAX_PORT_NUMBER)
17                 portNumber = sftp.DEFAULT_PORT_NUMBER;
18             if (connectTimeout <sftp.MIN_CONNECT_TIMEOUT) connectTimeout = sftp.MIN_CONNECT_TIMEOUT; else if (connectTimeout >
sftp.MAX_CONNECT_TIMEOUT)
19                 connectTimeout = sftp.MAX_CONNECT_TIMEOUT;
20
21             var connection = sftp.createConnection({
22                 username: 'sftpuser',
23                 keyId: 'custkey1',
24                 url: '192.168.0.100',
25                 port: portNumber,
26                 directory: 'inbound',
27                 timeout: connectTimeout,
28                 hostKey: "AAAAB3NzaC1yc2EAAAQABAAQDMifKH2vTxdiy8nem7+1S3x7dTQR/A67KdsR/5C2WUcDipBzYHb
nG6Am12Nd2t1M01LnaBZa6/8P4Y9x/sGTxtsdE/MzeGDUBn6HB1QvgIrhX62wgoKGQ+P21EA01+Vz8y3/MB1NmD7Fc62cJ9Mu88Y6jwJOIPZeHYNVYIm90iY6VyzYyvSJh
H0x7SXyyGnijQf4G8C4c8u/UvPf/sE16xKztly2Rx0aDL2FsDRtpyPmM602/R6ISbsmgab3MzzAEIu+zLDMdIBJn3cDhNt1F7Rar6Tu0u18KCkk8GPxbnxDuG4sCNOx
PYkDXSUmB/ocRjYGtqdZUMmeTf3"
29             });
30
31             // can also be a big file (created for example by async search)
32             var myFileToUpload = file.create({
33                 name: 'originalname.txt',
34                 fileType: file.Type.PLAINTEXT,
35                 contents: 'I am a test file.'
36             });
37
38             if (myFileToUpload.size > connection.MAX_FILE_SIZE)
39                 throw error.create({name:"FILE_IS_TOO_BIG", message:"The file you are trying to upload is too big"});
40
41             if (transferTimeout > connection.MAX_TRANSFER_TIMEOUT)
42                 transferTimeout = connection.MAX_TRANSFER_TIMEOUT;
43             else if (transferTimeout < connection.MIN_TRANSFER_TIMEOUT)
44                 transferTimeout = connection.MIN_TRANSFER_TIMEOUT;
45
46
47             connection.upload({
48                 directory: 'files',
49                 filename: 'test.txt',
50                 file: myFileToUpload,
51                 replaceExisting: true,
52                 timeout: transferTimeout
53             });
54
55         }
56     }
57 }
```

```
56 |     );
57 | });

```

Setting up an SFTP Transfer

- Development Preparation for SFTP transfers
- Execution of an SFTP transfer

Development Preparation for SFTP transfers

To successfully connect to your SFTP server with SuiteScript, the following steps are recommended:

1. Talk to your SFTP service provider about your plans.
 - Determine the connection properties required to connect with your external SFTP server. For example:
 - username
 - password/key
 - url
 - port
 - upload/download directories
 - host key
 - host key type
 - Make sure that you know your provider's practices around host key changes, maintenance and failover. For example, find out if there are multiple URLs or ports to try.
 - Check compatibility with the SFTP ciphers supported by NetSuite. See [Supported Cipher Suites and Host Key Types](#).
 - Determine if your provider requires at-rest file encryption (in addition to what the SFTP protocol provides during transfer). Decide if you need to add file encryption.
2. Build a credential management Suitelet to capture username and password token. Then, test the connection.
 - Create custom fields to store the user's SFTP username and password token
 - Implement the Suitelet.
 - a. Draw a form on a GET request.
 - b. Save the username and password token on a POST request.
 - c. Test the connection.

See [Creating a Suitelet Form that Contains a Credential Field](#).

- Build a server-side script to handle operations such as:
 - Load a File Cabinet file and upload it to the SFTP server.
 - Download an on demand file from the SFTP server and save it in File Cabinet.

Execution of an SFTP transfer

The following steps occur during a successful SFTP transfer using SuiteScript:

1. User submits their SFTP credentials via a Suitelet.

2. Suitelet captures and stores the credential token.
3. A server-side script is triggered.
4. Script identifies the appropriate credential token and other connection attributes, and establishes the SFTP connection.
5. Script requests the transfer.

SFTP Authentication

Please review the following sections for an overview of SFTP authentication when using SuiteScript.

- [Credential Tokenization](#)
- [Creating a Suitelet Form that Contains a Credential Field](#)
- [Reading the Credential Token in a Suitelet](#)
- [Credential Management](#)
- [Credential GUID Persistence](#)
- [Protocols](#)
- [Host Key Verification](#)
- [Retrieving the Host Key of an External SFTP Server](#)

Credential Tokenization

SuiteScript provides the ability for users to securely store authentication credentials in such a way that scripts are able to utilize encrypted saved credentials without being able to see their contents. The script author must specify which scripts and domains are permitted for use with the credential. To restrict the credential for use by SuiteScript automation triggered by the same user who originally saved the credential, the script author can set the `restrictToCurrentUser` parameter.

Creating a Suitelet Form that Contains a Credential Field

 **Note:** Credential fields have a default maximum length of 32 characters. If needed, use the [Field.maxLength](#) property to change this value

```

1 ...
2 if(request.method === context.Method.GET){
3   var form = ui.createForm({title: 'Enter SFTP Credentials'});
4   var credField = form.addCredentialField({
5     id: 'custfield_sftp_password_token',
6     label: 'SFTP Password',
7     restrictToScriptIds: ['customscript_sftp_script'],
8     restrictToDomains: ['acmebank.com'],
9     restrictToCurrentUser: true //Depends on use case
10   });
11   credField.maxLength = 64;
12   form.addSubmitButton();
13   response.writePage(form);
14 }
15 ...

```

Reading the Credential Token in a Suitelet

Note that the following code snippet is not a fully functional sample.

```

1 ...
2 var request = context.request;
3 if(request.method === context.Method.POST){
4     // Read the request parameter matching the field ID we specified in the form
5     var passwordToken = request.parameters.custfield_sftp_password_token;
6     log.debug({
7         title: 'New password token',
8         details: passwordToken
9     });
10    // In a real-world script, "passwordToken" is saved into a custom field here...
11 }
12 ...

```

Credential Management

User passwords can be stored using secure Credential Fields. This type of field is available on the [serverWidget.Form](#) Object in the [N/ui/serverWidget Module](#).

Encrypted *custom* fields do not support tokenization and are not compatible with the SFTP module. Instead, you can add a credential field using [Form.addCredentialField\(options\)](#).

Credential GUID Persistence

Scripts may store credential tokens as convenient for the script author. Credential tokens are not related to the password in its original or encrypted form within NetSuite. These tokens are unique identifiers which allow a script to refer to an encrypted secret securely stored within the SuiteCloud platform. Automatic password expiration is not currently provided, nor is it possible to view an inventory of saved credentials in the user interface.

Protocols

The SFTP module allows scripts to transfer files using the SSH File Transfer Protocol only. Other file-based protocols such as FTP, FTPS, SCP are not supported by this module.

Host Key Verification

An SFTP server identifies itself using a host key when a client attempts to establish a connection. Host keys are unique keys that the underlying SSH protocol uses to allow the server to provide a fingerprint. Clients can verify that the expected server has responded to the connection request for a particular URL and port number.

SuiteScript requires that the host key is provided by the script attempting to connect so that the SFTP module can check the identity of the SFTP server. This security best practice is commonly referred to as "Strict Host Key Checking".

Host keys are used to verify the identity of the server, not the client. For more information, see the help topic [SSH Keys for SFTP](#).

By design, there is no SuiteScript API call for checking the host key of a remote SFTP server, or an option to disable strict host key checking. The script must always know the host key ahead of time.

We recommend using OpenSSH's ssh-keyscan tool to check the host key of an external SFTP site. See [Retrieving the Host Key of an External SFTP Server](#) and [The OpenBSD's ssh-keyscan page](#).

Retrieving the Host Key of an External SFTP Server

An example usage checking the RSA host key of URL: acme.com at port: 1234 from a *nix shell follows:

```

1 | $ ssh-keyscan -t rsa -p 1234 acme.com
2 | AATpn1P9jB+cQx9Jjq9UeZjA1245X7SB0cR1Kh+Sok56VzSw==
```

It is recommended to always pass the key type and port number. This practice helps to avoid ambiguity in the response from the external SFTP server.

Supported Cipher Suites and Host Key Types

SFTP connections are encrypted. For security reasons, NetSuite requires that the server to which a connection request is being made supports at least one of the following ciphers: aes128-ctr, aes192-ctr or aes256-ctr. The preceding cipher specs refer to the AES cipher in Counter stream cipher mode using 128, 192 or 256 bit key sizes.

To check interoperability of your SFTP server or service provider, refer to the following table:

Communication protocol	<p>SFTP (SSH + FTP) is supported.</p> <p>Only CTR (and not CBC) ciphers are allowed. Your SFTP server can use the following encryption algorithms:</p> <ul style="list-style-type: none"> ■ AES 128-CTR ■ AES 192-CTR ■ AES 256-CTR ■ RSA ■ DSA ■ ECDSA <p>Files are not additionally encrypted during transfer. The entire transmission is encrypted by the SSH protocol.</p>
Authentication mechanism	<p>Username</p> <p>Password</p> <p>Password/SSH key with or without passphrase</p>
SSH host key	With each connection request, you must supply the host key. Any host key changes need to be managed manually.
GUID	<p>The password GUID should be a value generated by a credential field from a Suitelet using Form.addCredentialField(options).</p> <p>The password GUID field's originating credential field must include the SFTP domain on the <code>restrictToDomains</code> parameter.</p> <p>The password GUID field's originating credential field must include the script utilizing the password GUID on the <code>restrictToScriptIds</code> parameter.</p>

Firewall policy is at the discretion of your SFTP service provider.

Supported SuiteScript File Types

SuiteScript has two types of file objects: previously existing files in the NetSuite File Cabinet, and on demand files created using SuiteScript API calls such as [file.create\(options\)](#) or [Connection.download\(options\)](#).

File Cabinet and on demand files are supported by [Connection.upload\(options\)](#).

Note that `Connection.download(options)` returns an on demand file object. For an on demand file to be saved into the File Cabinet, it must receive a folder ID and be explicitly saved.

```

1 ...
2 var downloadedFile = sftp.download({...});
3 downloadedFile.folder = 1234;
4 downloadedFile.save();
5 ...

```



Important: It's possible that a file you are downloading may be encrypted, or your SFTP provider may expect an uploaded file in a encrypted format in accordance with that provider's security practices. Make sure that you understand your provider's expectations and the cryptographic capabilities in SuiteScript (see [N/crypto Module](#)).

You can also create and remove directories. For more information, see [N/sftp Module Members](#).

Syntax

```

1 require(['N/sftp', 'N/file'],
2   function (sftp, file)
3   {
4
5     var connection = sftp.createConnection({
6       /*
7         The Username supplied by the administrator of the external SFTP server.
8         */
9       username: 'myuser',
10      /*
11        Refers to the Password supplied by the administrator of the external SFTP server.
12
13        The Password Token/GUID obtained by reading the form POST parameter associated
14        with user submission of a form containing a Credential Field.
15
16        Value would typically be read from a custom field.
17        */
18       passwordGUID: "B34672495064525E5D65032D63B52301",
19      /*
20        The URL supplied by the administrator of the external SFTP server.
21        */
22       url: 'host.somewhere.com',
23      /*
24        The SFTP Port number supplied by the administrator of the external SFTP server (defaults to 22).
25        */
26       port: 22,
27      /*
28        The transfer directory supplied by the administrator of the external SFTP server (optional).
29        */
30       directory: 'transferfiles',
31      /*
32        RSA Host Key obtained via ssh-keyscan tool.
33
34        $ ssh-keyscan -t rsa -p 22 host.somewhere.com
35        AATpn1P9jB+cQx9Jq9UeZjA1245X7SBDCRiKh+Sok56VzSw==
36        /*
37        hostKey: "AATpn1P9jB+cQx9Jq9UeZjA1245X7SBDCRiKh+Sok56VzSw=="
38      });
39
40      /*
41        Creating a simple file.
42        */
43       var myFileToUpload = file.create({
44         name: 'originalname.js',
45         fileType: file.Type.PLAINTEXT,
46         contents: 'I am a test file. Hear me roar.'
47       });
48

```

```

49  /*
50   * Uploading the file to the external SFTP server.
51   */
52   connection.upload({
53     /*
54      Subdirectory within the transfer directory specified when connecting (optional).
55      */
56      directory: 'relative/path/to/remote/dir',
57      /*
58        Alternate file name to use instead of the one given to the file object (optional).
59        */
60      filename: 'newFileNameOnServer.js',
61      /*
62        The file to upload.
63        */
64      file: myFileToUpload,
65      /*
66        If a file already exists with that name, replace it instead of failing the upload.
67        */
68      replaceExisting: true
69    });
70
71  var downloadedFile = connection.download({
72    /*
73      Subdirectory within the transfer directory specified when connecting (optional).
74      */
75      directory: 'relative/path/to/file',
76      /*
77        The name of the file within the above directory on the external SFTP server which to download.
78        */
79      filename: 'downloadMe.js'
80    });
81
82  });
83

```

sftp.Connection



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Represents a connection to the account on the remote FTP server.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/sftp Module
Methods and Properties	Connection Object Members
Since	2016.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```

1 //Add additional code ...
2 // establish connection to the FTP server
3 var objConnection = sftp.createConnection({
4   username: 'username',
5   keyId: 'custkey1',
6   url: 'host.somewhere.com',
7   directory: 'username/wheres/my/file'

```

```

8 |     hostKey: myHostKey
9 | });
10| ...
11| //Add additional code

```

Connection.download(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Downloads a file from the remote FTP server.
Returns	file.File Object
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	100 units
Module	N/sftp Module
Since	2016.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.filename	string	Required	The name of the file to download.	2016.2
options.directory	string	Optional	<p>The relative path to the directory that contains the file to download.</p> <p>By default, the path is set to the current directory.</p> <p>Important: This input must take the form of a relative path.</p>	2016.2
options.timeout	number	Optional	<p>The number of seconds to allow for the file to download.</p> <p>By default, this value is set to 300 seconds.</p>	2016.2

Errors

Error Code	Thrown If
FTP_MAXIMUM_FILE_SIZE_EXCEEDED	The file size is greater than the maximum file size allowed by NetSuite.
FTP_NO_SUCH_FILE_OR_DIRECTORY	The file or directory does not exist.
FTP_TRANSFER_TIMEOUT_EXCEEDED	The transfer is taking longer than the specified options.timeout value.

Error Code	Thrown If
FTP_INVALID_TRANSFER_TIMEOUT	The options.timeout value is either a negative value, zero or greater than 300 seconds.
FTP_PERMISSION_DENIED	Access to the file or directory on the remote FTP server was denied.
CONNECTION_RESET	The connection was reset.
THE_REMOTE_PATH_FOR_FILE_IS_NOT_VALID	The file's remote path is invalid.
CONNECTION_CLOSED_BY_HOST	The connection was closed by the host.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var downloadedFile = objConnection.download({
4   directory: 'relative/path/to/file',
5   filename: 'downloadMe.js'
6 });
7 ...
8 //Add additional code

```

Connection.upload(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Uploads a file to the remote FTP server. The maximum file size that can be uploaded to is 100 MB.
Returns	void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	100 units
Module	N/sftp Module
Since	2016.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.file	file.File	Required	The file to upload.	2016.2
options.filename	string	Optional	The name to give the uploaded file on server.	2016.2

Parameter	Type	Required / Optional	Description	Since
			<p>By default, the filename is the same specified by options.filename.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> i Note: Illegal characters are automatically escaped. </div>	
options.directory	string	Optional	<p>The relative path to the directory where the file should be upload to.</p> <p>By default, the path is set to the current directory.</p> <div style="border: 1px solid #FFD700; padding: 5px; margin-top: 10px;"> ! Important: This input must take the form of a relative path. </div>	2016.2
options.timeout	number	Optional	<p>The number of seconds to allow for the file to upload.</p> <p>By default, this value is set to 300 seconds.</p> <div style="border: 1px solid #FFD700; padding: 5px; margin-top: 10px;"> ! Important: This parameter does not specify the overall timeout limit. The value is only applied when no data is received within the specified period. </div>	2016.2
options.replaceExisting	boolean true false	Optional	<p>Indicates whether the file being uploaded should overwrite any file with the name options.filename that already exists in options.directory.</p> <p>If false, the FTP_FILE_ALREADY_EXISTS exception is thrown when a file with the same name already exists in the options.directory.</p> <p>By default, this value is false.</p>	2016.2

Errors

Error Code	Thrown If
FTP_NO_SUCH_FILE_OR_DIRECTORY	The file or directory does not exist.
FTP_TRANSFER_TIMEOUT_EXCEEDED	The transfer is taking longer than the specified options.timeout value.
FTP_INVALID_TRANSFER_TIMEOUT	The options.timeout value is either a negative value, zero or greater than 300 seconds.
FTP_FILE_ALREADY_EXISTS	The options.replaceExisting value is false and a file with the same name exists in the remote directory.
CONNECTION_RESET	The connection was reset.
THE_REMOTE_PATH_FOR_FILE_IS_NOT_VALID	The file's remote path is invalid.
CONNECTION_CLOSED_BY_HOST	The connection was closed by the host.

Error Code	Thrown If
FTP_PERMISSION_DENIED	Access to the file or directory on the remote FTP server was denied.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```

1 //Add additional code ...
2 objConnection.upload({
3   directory: 'relative/path/to/remote/dir',
4   filename: 'newFileNameOnServer',
5   file: myFileToUpload,
6   replaceExisting: true
7 });
8 ...
9 //Add additional code

```

Connection.makeDirectory(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates an empty directory.
Returns	void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/sftp Module
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.path	string	Required	The relative path of a directory to be created.	2019.2

Errors

Error Code	Thrown If
FTP_PERMISSION_DENIED	Access to the file or directory on the remote FTP server was denied.

Error Code	Thrown If
FTP_DIRECTORY_NOT_FOUND	Creating a directory in a non-existent directory.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```

1 // Add additional code ...
2 objConnection.makeDirectory({
3     directory: 'relative/path/to/remote/dir',
4 });
5 ...
6 //Add additional code

```

Connection.removeDirectory(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes an empty directory.
Returns	void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/sftp Module
Since	2019.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.path	String	Required	The relative path of a directory to be deleted.	2019.2

Errors

Error Code	Thrown If
FTP_PERMISSION_DENIED	Access to the file or directory on the remote FTP server was denied.
FTP_DIRECTORY_NOT_FOUND	Deleting a directory that does not exist.
FTP_DIRECTORY_NOT_EMPTY	Deleting a directory that is not empty.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```

1 //Add additional code ...
2 objConnection.removeDirectory({
3   directory: 'relative/path/to/remote/dir',
4 });
5 ...
6 //Add additional code

```

Connection.removeFile(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes a file.
Returns	void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/sftp Module
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.path	String	Required	The relative path of the directory location where this file should be removed.	2019.2

Errors

Error Code	Thrown If
FTP_PERMISSION_DENIED	Access to the file or directory on the remote FTP server was denied.
FTP_NO_SUCH_FILE_OR_DIRECTORY	The file or directory does not exist.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```
1 //Add additional code ...
```

```

2 objConnection.removeFile({
3   directory: 'relative/path,
4   });
5 ...
6 //Add additional code

```

Connection.move(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Moves a file or directory from one location to another.
Returns	void
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/sftp Module
Since	2019.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.from	String	Required	The relative path of the file to be moved from.	2019.2
options.to	String	Required	The relative path of the file to be moved to.	2019.2

Errors

Error Code	Thrown If
FTP_INVALID_MOVE	Source is not readable or the target is not writable. Source or target does not exist.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```

1 // Add additional code ...
2 objConnection.move({
3   from: 'relative/path/to/remote/dir',
4   to: 'relative/path/to/remote/dir/new',
5   });
6 ...
7 // Add additional code

```

Connection.list(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Lists the remote directory.</p> <p>This method returns an array of Objects. Each Object includes the following properties with associated values:</p> <ul style="list-style-type: none"> ▪ directory- A flag whether the entry corresponds to a directory or a file. If true, it is a directory. If false, it is a file. ▪ name- The name of the file ▪ size- The size of the file ▪ lastModified- The last modification date
Returns	Array<Object>
Supported Script Types	<p>Server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	10 units
Module	N/sftp Module
Since	2019.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.path	String	Required	The relative path to directory of file that will be downloaded.	2019.2
options.sort	String	Required	The sort options. Values from the sort enum are accepted.	2019.2

Errors

Error Code	Thrown If
FTP_INVALID_DIRECTORY	The directory does not exist on the remote FTP server.
FTP_PERMISSION_DENIED	Access to the file or directory on the remote FTP server was denied.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```

1 // Add additional code...
2 var connection = sftp.createConnection({
```

```

3 |     username: 'sftpuser',
4 |     keyId:"custkey2",
5 |     url: '172.25.184.111',
6 |     port: 22,
7 |     directory: '',
8 |     hostKey: "hostkey"
9 |   });
10 // Add additional code

```



Note: Wildcards are accepted. The ? symbol can represent any character. The * symbol can represent any number of characters.

Connection.MAX_FILE_SIZE



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the maximum file size.
Type	enum
Module	N/sftp Module
Sibling Module Members	N/sftp Module Members
Since	2019.2

Values

- 100000000

Syntax

See [N/sftp Module Script Samples](#).

Connection.MAX_TRANSFER_TIMEOUT



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the maximum transfer timeout.
Type	enum
Module	N/sftp Module
Sibling Module Members	N/sftp Module Members

Since	2019.2
-------	--------

Values

- 300

Syntax

See [N/sftp Module Script Samples](#).

sftp.createConnection(options)

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Method Description	Establishes a connection to a remote FTP server. To generate the passwordguid, you can create a suitelet that uses Form.addCredentialField(options) . Use the N/https Module to fetch the GUID value returned from the Suitelet's credential field. For more information, see Setting up an SFTP Transfer and Supported Cipher Suites and Host Key Types .
Returns	sftp.Connection , representing that connection.
Supported Script Types	All server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/sftp Module
Since	2016.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.url	string	Required	The host of the remote account.	2016.2
options.passwordGuid	string	Required	The password GUID for the remote account. This is only required if key ID is not provided.	2016.2
options.hostKey	string	Required	The host key for the trusted fingerprint on the server. The host key is required even if keyId is supplied instead of passwordGuid.	2016.2
options.username	string	Required	The username of the remote account.	2016.2

Parameter	Type	Required / Optional	Description	Since
			By default, the login is anonymous.	
options.port	number	Optional	The port used to connect to the remote account. By default, port 22 is used.	2016.2
options.directory	string	Optional	The remote directory of the connection. Note: The directory property is required if you use a remote server cannot resolve relative paths.	2016.2
options.timeout	number	Optional	The number of seconds to allow for an established connection. By default, this value is set to 20 seconds.	2016.2
options.hostKeyType	string	Optional	The type of host key specified by options.hostKey. This value can be set to one of the following options: <ul style="list-style-type: none">■ dsa■ ecdsa■ rsa	2016.2
options.keyId	string	Optional	The ID of the key to be used for authentication.	2016.2

Errors

Error Code	Thrown If
FTP_UNKNOWN_HOST	The host could not be found.
FTP_CONNECT_TIMEOUT_EXCEEDED	A connection could not be established within options.timeout seconds.
FTP_CANNOT_ESTABLISH_CONNECTION	The password/username was invalid or permission to access the directory was denied.
FTP_INVALID_PORT_NUMBER	The port number is invalid.
FTP_INVALID_CONNECTION_TIMEOUT	The options.timeout value is either a negative value, zero, or greater than 20 seconds.
FTP_INVALID_DIRECTORY	The directory does not exist on the remote FTP server.
FTP_INCORRECT_HOST_KEY	The host key does not match the presented host key on the remote FTP server.
FTP_INCORRECT_HOST_KEY_TYPE	The host key type and provided host key type do not match.
FTP_MALFORMED_HOST_KEY	The host key is not in the correct format. (e.g. base 64, 96+ bytes)

Error Code	Thrown If
FTP_PERMISSION_DENIED	Access to the file or directory on the remote FTP server was denied.
FTP_UNSUPPORTED_ENCRYPTION_ALGORITHM	The remote FTP server does not support one of NetSuite's approved algorithms. (e.g. aes256-ctr, es192-ctr, es128-ctr)
AUTHENTICATION_FAIL_TOO_MANY_INCORRECT_AUTHENTICATION_ATTEMPTS	There are too many incorrect authentication attempts.
NO_ROUTE_TO_HOST_FOUND	No route to the host can be found.
CONNECTION_RESET	The connect was reset.
CONNECTION_CLOSED_BY_HOST	The connection was closed by the host.
THE_REMOTE_PATH_FOR_FILE_IS_NOT_VALID	The file's remote path is invalid.
SFTPCREDENTIAL_ENCODING_ERROR	There is an SFTP credential encoding error.
UNABLE_TO_GET_SFTP_SERVER_ADDRESS	The SFTP server address is unavailable.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/sftp Module Script Samples](#).

```

1 //Add additional code
2 ...
3 // establish connection to the ftp server
4
5 var objConnection = sftp.createConnection({
6   username: 'username',
7   keyId: 'custkey1',
8   url: 'host.somewhere.com',
9   directory: 'username/wheres/my/file'
10  hostKey: myHostKey // references var myHostKey
11 });
12 ...
13 //Add additional code

```

sftp.MAX_CONNECT_TIMEOUT

i Note: The content in this help topic pertains to SuiteScript 2.0.

i Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the maximum connection timeout.
Type	enum
Module	N/sftp Module
Sibling Module Members	N/sftp Module Members

Since	2016.2
-------	--------

Values

- 20

Syntax

See [N/sftp Module Script Samples](#).

sftp.MIN_CONNECT_TIMEOUT

 Note:	The content in this help topic pertains to SuiteScript 2.0.
 Note:	JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the minimum connection timeout.
Type	enum
Module	N/sftp Module
Sibling Module Members	N/sftp Module Members
Since	2016.2

Values

- 1

Syntax

See [N/sftp Module Script Samples](#).

sftp.MAX_PORT_NUMBER

 Note:	The content in this help topic pertains to SuiteScript 2.0.
 Note:	JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the maximum port number.
Type	enum

Module	N/sftp Module
Sibling Module Members	N/sftp Module Members
Since	2019.2

Values

- 65535

Syntax

See [N/sftp Module Script Samples](#).

sftp.MIN_PORT_NUMBER

Note: The content in this help topic pertains to SuiteScript 2.0.
Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for the minimum port number.
Type	enum
Module	N/sftp Module
Sibling Module Members	N/sftp Module Members
Since	2019.2

Values

- 0

Syntax

See [N/sftp Module Script Samples](#).

sftp.DEFAULT_PORT_NUMBER

Note: The content in this help topic pertains to SuiteScript 2.0.
Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values for default port number.
-------------------------	---

Type	enum
Module	N/sftp Module
Sibling Module Members	N/sftp Module Members
Since	2019.2

Values

- 22

Syntax

See [N/sftp Module Script Samples](#).

sftp.Sort

 Note:	The content in this help topic pertains to SuiteScript 2.0.
 Note:	JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the values to be used to sort the listed directory.
Type	enum
Module	N/sftp Module
Sibling Module Members	N/sftp Module Members
Since	2019.2

Values

- DATE
- DATE_DESC
- SIZE
- SIZE_DESC
- NAME
- NAME_DESC

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/sftp Module Script Samples .
---	--

```
1 | //Add additional code ...
```

```

2 require(['N/sftp'],function(sftp){
3     var connection = sftp.createConnection({
4         username: 'sftpuser',
5         keyId: 'custkey1',
6         url: '192.168.0.100',
7         port: 22,
8         directory: 'inbound',
9         hostKey: "AAAAAB3NzaC1yc2EAAAQABAAQDMifKH2vTxdiy8neM7+1S3x7dTQR/A67KdsR/5C2WUcDipBzYhHb
nG6Am12Nd2t1M01LnaBzA6/8P4Y9x/sGTxtsdE/MzeGDUBn6HB1QvgIrhX62wgoKGQ+P21EA01+Vz8y3/MB1NmD7Fc62cJ9Mu88YA6jwJOIPZeHYNV90rY6VyzYyvSJh
H0x7SxyvGnijQF4G8C4c8u/UVpf/sE16xKZtly2Rx0aDL2FsDRtpyPmM602/R6ISbsmgab3MzzAEIu+zLDMdIBJn3cDhNt1F7Rar6Tu0u18CKkk8GPxbnxDu4sCNOoX
PYkDXSMUbM/ocRjYGtqdZUMmeTf3"
10    });
11    connection.list({
12        path:"?path*",
13        sort: sftp.Sort.SIZE});
14 })
15 //Add additional code

```

N/sso Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Use the sso module to generate outbound single sign-on (SuiteSignOn) tokens. For example, to create a reference to a SuiteSignOn record, or to integrate with an external application.

For more information about the SuiteSignOn feature, see the help topic [Outbound Single Sign-on \(SuiteSignOn\)](#).

- [N/sso Module Member](#)
- [N/sso Module Script Sample](#)

N/sso Module Member

Member Type	Name	Return Type	Supported Script Types	Description
Method	sso.generateSuiteSignOnToken(options)	string	Portlet scripts, user event scripts, and Suitelets	Generates a new SuiteSignOn token for a user

N/sso Module Script Sample



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).



Important: The value used in this sample for the suiteSignOnRecordId field is a placeholder. Before using this sample, replace the suiteSignOnRecordId field value with a valid value from your NetSuite account. If you run a script with an invalid value, an error may occur. Additionally, the SuiteSignOn record you reference must be associated with a specific script. You make this association in the SuiteSignOn record's Connection Points sublist. For help with SuiteSignOn records, see the help topic [Creating SuiteSignOn Records](#).

The following sample shows how to generate a new OAuth token for a user. This sample requires the SuiteSignOn feature.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/sso'],function(sso) {
6      function generateSSOToken() {
7          var suiteSignOnRecordId = 1;
8          var url = sso.generateSuiteSignOnToken(suiteSignOnRecordId);
9      }
10     generateSSOToken();
11 });

```

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to use generateSuiteSignOnToken(options) in a portlet script.

Important: The value used in this sample for the suiteSignOnRecordId field is a placeholder. Before using this sample, replace the suiteSignOnRecordId field value with a valid value from your NetSuite account. If you run a script with an invalid value, an error may occur. Additionally, the SuiteSignOn record you reference must be associated with a specific script. You make this association in the SuiteSignOn record's Connection Points sublist. For help with SuiteSignOn records, see the help topic [Creating SuiteSignOn Records](#).

```

1  /**
2   * @NApiVersion 2.0
3   * @NScriptType Portlet
4   * @NScriptPortletType form
5   */
6
7  define(['N/sso'], function (sso) {
8      function render(context) {
9          var suiteSignOnRecordId = 'customsso_test';
10         var url = sso.generateSuiteSignOnToken(suiteSignOnRecordId);
11         log.debug(url);
12     }
13     return {
14         render: render
15     };
16 });

```

Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to use generateSuiteSignOnToken(options) in a Suitelet script.

Important: The value used in this sample for the suiteSignOnRecordId field is a placeholder. Before using this sample, replace the suiteSignOnRecordId field value with a valid value from your NetSuite account. If you run a script with an invalid value, an error may occur. Additionally, the SuiteSignOn record you reference must be associated with a specific script. You make this association in the SuiteSignOn record's Connection Points sublist. For help with SuiteSignOn records, see the help topic [Creating SuiteSignOn Records](#).

```

1  /**
2   * @NApiVersion 2.x

```

```

3  * @NScriptType Suitelet
4  */
5 define(['N/sso'], function(sso) {
6     function onRequest(context) {
7         var suiteSignOnRecordId = 'customsso_test'; //Replace placeholder values
8         var url = sso.generateSuiteSignOnToken(suiteSignOnRecordId);
9         log.debug(url);
10    }
11
12    return {
13        onRequest: onRequest
14    };
15 });

```



Note: This script sample uses the `define` function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the `require` function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample shows how to use `generateSuiteSignOnToken(options)` in a user event script.



Important: The value used in this sample for the `suiteSignOnRecordId` field is a placeholder. Before using this sample, replace the `suiteSignOnRecordId` field value with a valid value from your NetSuite account. If you run a script with an invalid value, an error may occur. Additionally, the SuiteSignOn record you reference must be associated with a specific script. You make this association in the SuiteSignOn record's Connection Points sublist. For help with SuiteSignOn records, see the help topic [Creating SuiteSignOn Records](#).

```

1 /**
2  * @NApiVersion 2.0
3  * @NScriptType UserEventScript
4  * @NModuleScope SameAccount
5 */
6
7 define(['N/sso'], function(sso) {
8     function beforeLoad(context) {
9         var suiteSignOnRecordId = 'customsso_test';
10        var url = sso.generateSuiteSignOnToken(suiteSignOnRecordId);
11        log.debug(url);
12    }
13    return {
14        beforeLoad: beforeLoad
15    };
16 });

```

sso.generateSuiteSignOnToken(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Method used to generate a new SuiteSignOn token for a user.</p> <p>Note: To use this method, Outbound Single Sign-on and SOAP web services must be enabled in your account. To enable these features, go to Setup > Company > Enable Features. On the SuiteCloud tab, in the Manage Authentication section, select the SuiteSignOn check box. In the SuiteTalk section, select the SOAP Web Services check box. Click Save.</p>
Returns	URL, OAuth token, and any integration variables as a string

Supported Script Types	Portlet scripts, user event scripts, and Suitelets For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/sso Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.suiteSignOnId	string	required	The scriptId specified on the SuiteSignOn record. To see a list of IDs for SuiteSignOn records, go to the SuiteSignOn list page (Setup > Integration > SuiteSignOn). i Note: NetSuite recommends that you create a custom scriptId for each SuiteSignOn record to avoid naming conflicts should you decide use SuiteBundler to deploy your scripts into other accounts.	2015.2

Errors

Error Code	Message	Thrown If
INVALID_SSO	Invalid SuiteSignOn reference: {1}. That SuiteSignOn object does not exist or has been marked as inactive.	The suiteSignOnId input parameter is invalid or does not exist. i Note: The suiteSignOnId input parameter must be a scriptId and not a internal id.
SSO_CONFIG_REQD	The SuiteSignOn object {1} is not configured for use with this script. You must specify the script as a connection point for this SuiteSignOn.	The suiteSignOnId input parameter is missing.

Syntax

! Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/sso Module Script Sample .

```

1 //Add additional code
2 ...

```

```

3 | var suiteSignOnRecordId = 1;
4 | var url = sso.generateSuiteSignOnToken('customsso1');
5 | ...
6 | //Add additional code

```

N/task Module

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Load the task module to create tasks and place them in the internal NetSuite scheduling or task queue. Use the task module to do the following:

- Schedule scripts
- Run map/reduce scripts
- Import CSV files
- Merge duplicate records
- Execute asynchronous searches, constructed queries, SuiteQL queries, and workflows

Each task type has its own corresponding object types. Use the methods available to each object type to configure, submit, and monitor the tasks.

 **Note:** Regardless of task type, tasks are always triggered asynchronously.

- N/task Module Members
- CsvImportTask Object Members
- CsvImportTaskStatus Object Members
- EntityDeduplicationTask Object Members
- EntityDeduplicationTaskStatus Object Members
- MapReduceScriptTask Object Members
- MapReduceScriptTaskStatus Object Members
- QueryTask Object Members
- QueryTaskStatus Object Members
- RecordActionTask Object Members
- RecordActionTaskStatus Object Members
- ScheduledScriptTask Object Members
- ScheduledScriptTaskStatus Object Members
- SearchTask Object Members
- SearchTaskStatus Object Members
- SuiteQLTask Object Members
- SuiteQLTaskStatus Object Members
- WorkflowTriggerTask Object Members
- WorkflowTriggerTaskStatus Object Members

- N/task Module Script Samples

N/task Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	task.CsvImportTask	Object	Server scripts	The properties of a CSV import task. Use the methods and properties for this object to submit a CSV import task into the task queue and asynchronously import record data into NetSuite.
	task.CsvImportTaskStatus	Object	Server scripts	The status of a CSV import task placed into the NetSuite scheduling queue.
	task.EntityDeduplicationTask	Object	Server scripts	All the properties of a merge duplicate records task request. Use the methods and properties of this object to submit a merge duplicate record job task into the NetSuite task queue.
	task.EntityDeduplicationTaskStatus	Object	Server scripts	The status of a merge duplicate record task placed into the NetSuite task queue.
	task.MapReduceScriptTask	Object	Server scripts	A map/reduce script deployment.
	task.MapReduceScriptTaskStatus	Object	Server scripts	The status of a map/reduce script deployment that has been submitted for processing.
	task.RecordActionTask	Object	Server scripts	The properties of a record action task. Use this object to place a record action task into the NetSuite scheduling queue.
	task.RecordActionTaskStatus	Object	Server scripts	The status of a record action task in the NetSuite scheduling queue.
	task.ScheduledScriptTask	Object	Server scripts	All the properties of a scheduled script task in SuiteScript. Use this object to place a scheduled script deployment into the NetSuite scheduling queue.
	task.ScheduledScriptTaskStatus	Object	Server scripts	The status of a scheduled script placed into the NetSuite scheduling queue.
	task.SearchTask	Object	Server scripts	The properties required to initiate an asynchronous search.
	task.SearchTaskStatus	Object	Server scripts	The status of an asynchronous search initiation task that is placed into the NetSuite task queue.
	task.WorkflowTriggerTask	Object	Server scripts	All the properties required to asynchronously initiate a workflow. Use WorkflowTriggerTask to create a task that initiates an instance of a specific workflow.
	task.WorkflowTriggerTaskStatus	Object	Server scripts	The status of an asynchronous workflow initiation task placed into the NetSuite task queue.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	task.checkStatus(options)	task.CsvImportTaskStatus task.EntityDeduplicationTaskStatus task.MapReduceScriptTaskStatus task.RecordActionTaskStatus task.ScheduledScriptTaskStatus task.SearchTaskStatus task.WorkflowTriggerTaskStatus	Server scripts	Returns a task status object associated with a specific task ID.
	task.create(options)	task.CsvImportTask task.EntityDeduplicationTask task.MapReduceScriptTask task.RecordActionTask task.ScheduledScriptTask task.SearchTask task.WorkflowTriggerTask	Server scripts	Creates an object for a specific task type and returns the task object.
Enum	task.ActionCondition	enum	Server scripts	Holds the string values for the possible record action conditions.
	task.DedupeEntityType	enum	Server scripts	Holds the string values for entity types for which you can merge duplicate records with task.EntityDeduplicationTask .
	task.DedupeMode	enum	Server scripts	Holds the string values for available deduplication modes when merging duplicate records with task.EntityDeduplicationTask .
	task.MapReduceStage	enum	Server scripts	Holds the string values for the stages of a map/reduce script deployment, which is encapsulated by the task.MapReduceScriptTask object.
	task.MasterSelectionMode	enum	Server scripts	Holds the string values for supported master selection modes when merging duplicate records with task.EntityDeduplicationTask .
	task.TaskStatus	enum	Server scripts	Holds the string values for the possible status of tasks created and submitted with the N/task Module .
	task.TaskType	enum	Server scripts	Holds the string values for the types of task objects, supported by the N/task Module , that you can create with task.create(options) .

CsvImportTask Object Members

The following members are called on [task.CsvImportTask](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	CsvImportTask.submit()	string	Server scripts	Directs NetSuite to place a CSV import task into the NetSuite task queue and returns a unique ID for the task.
Property	CsvImportTask.id	string	Server scripts	The ID of the task.
	CsvImportTask.importFile	file.File string	Server scripts	CSV file to import. Use a file.File object or a string that represents the CSV text to be imported.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	CsvImportTask.linkedFiles	Object	Server scripts	A map of key/value pairs that sets the data to be imported in a linked file for a multi-file import job, by referencing a file in the File Cabinet or the raw CSV data to import.
	CsvImportTask.mappingId	number string	Server scripts	Script ID or internal ID of the saved import map that you created when you ran the Import Assistant.
	CsvImportTask.name	string	Server scripts	Name for the CSV import task.
	CsvImportTask.queueId	number	Server scripts	Overrides the Queue Number property under Advanced Options on the Import Options page of the Import Assistant.

CsvImportTaskStatus Object Members

The following members are called on [task.CsvImportTaskStatus](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	CsvImportTaskStatus.status	string (read-only)	Server scripts	Status for a CSV import task. Returns a task.TaskStatus enum value.
	CsvImportTaskStatus.taskId	string (read-only)	Server scripts	The task ID associated with the specified task.

EntityDeduplicationTask Object Members

The following members are called on [task.EntityDeduplicationTask](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	EntityDeduplicationTask.submit()	string	Server scripts	Directs NetSuite to place the merge duplicate records task into the NetSuite task queue and returns a unique ID for the task.
Property	EntityDeduplicationTask.dedupeMode	string	Server scripts	The mode in which to merge or delete duplicate records. Use values from the task.DedupeMode enum.
	EntityDeduplicationTask.entityType	string	Server scripts	The type of entity on which you want to merge duplicate records. Use a task.DedupeEntityType enum to set the value.
	EntityDeduplicationTask.id	string	Server scripts	The ID of the task.
	EntityDeduplicationTask.masterRecordId	number	Server scripts	Master record ID. When you merge duplicate records, you can delete all duplicates for a record or merge

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				information from the duplicate records into the master record.
	EntityDeduplicationTask.masterSelectionMode	string	Server scripts	Master selection mode. Use values from the task.MasterSelectionMode enum.
	EntityDeduplicationTask.recordIds	number[]	Server scripts	Number array of record internal IDs to perform the merge or delete operation on.

EntityDeduplicationTaskStatus Object Members

The following members are called on [task.EntityDeduplicationTaskStatus](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	EntityDeduplicationTaskStatus.status	string (read-only)	Server scripts	Status for a merge duplicate record task. Returns a task.TaskStatus enum value.
	EntityDeduplicationTaskStatus.taskId	string (read-only)	Server scripts	The task ID associated with the specified task.

MapReduceScriptTask Object Members

The following members are called on [task.MapReduceScriptTask](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	MapReduceScriptTask.submit()	string	Server scripts	Submits a map/reduce script deployment for processing.
Property	MapReduceScriptTask.deploymentId	string	Server scripts	Script ID (as a string), for the script deployment record for a map/reduce script.
	MapReduceScriptTask.id	string	Server scripts	The ID of the task.
	MapReduceScriptTask.params	Object	Server scripts	Object that represents key/value pairs that override static script parameter field values on the script deployment record.
	MapReduceScriptTask.scriptId	number string	Server scripts	Internal ID (as a number), or script ID (as a string), for the map/reduce script record.

MapReduceScriptTaskStatus Object Members

The following members are called on the [task.MapReduceScriptTaskStatus](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	MapReduceScriptTaskStatus.getCurrentTotalSize()	number	Server scripts	Returns the total size in bytes of all stored work in progress by a task.MapReduceScriptTask .

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	MapReduceScriptTaskStatus.getPendingMapCount()	number	Server scripts	Returns the total number of records or rows not yet processed by the map stage of a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getPendingMapSize()	number	Server scripts	Returns the total number of bytes not yet processed by the map stage, as a component of total size, of a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getPendingOutputCount()	number	Server scripts	Returns the total number of records or rows not yet processed by a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getPendingOutputSize()	number	Server scripts	Returns the total size in bytes of all key/value pairs written as output, as a component of total size, by a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getPendingReduceCount()	number	Server scripts	Returns the total number of records or rows not yet processed by the reduce stage of a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getPendingReduceSize()	number	Server scripts	Returns the total number of bytes not yet processed by the reduce stage, as a component of total size, of a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getPercentageCompleted()	number	Server scripts	Returns the current percentage complete for the current stage of a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getTotalMapCount()	number	Server scripts	Returns the total number of records or rows passed as input to the map stage of a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getTotalOutputCount()	number	Server scripts	Returns the total number of records or rows passed as inputs to the output phase of a task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.getTotalReduceCount()	number	Server scripts	Returns the total number of record or row inputs to the reduce stage of a task.MapReduceScriptTask.
Property	MapReduceScriptTaskStatus.deploymentId	string (read-only)	Server scripts	Script ID for a script deployment record associated with a specific task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.scriptId	number (read-only)	Server scripts	Internal ID for a map/reduce script record associated with a specific task.MapReduceScriptTask.
	MapReduceScriptTaskStatus.stage	string (read-only)	Server scripts	The current stage of a map/reduce script deployment that is being processed. See task.MapReduceStage for supported values.
	MapReduceScriptTaskStatus.status	string (read-only)	Server scripts	Status for a map/reduce script task. Returns a task.TaskStatus enum value.
	MapReduceScriptTaskStatus.taskId	string (read-only)	Server scripts	The task ID associated with the specified task.

QueryTask Object Members

The following members are called on `task.QueryTask`.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	QueryTask.addInboundDependency(options)	void	Server scripts	Adds a scheduled script task or map/reduce script task to the query task as a dependent task.
	QueryTask.submit()	string	Server scripts	Submits the query task for asynchronous processing and returns the task ID.
Property	QueryTask.fileId	number	Server scripts	Internal ID of the CSV file to export query results to. This property is mutually exclusive with the QueryTask.filePath parameter.
	QueryTask.filePath	string	Server scripts	Path of the CSV file to export query results to. This property is mutually exclusive with the QueryTask.fileId property.
	QueryTask.inboundDependencies	Object[]	Server scripts	Key-value pairs that contain information about the dependent tasks added to the query task.
	QueryTask.query	string	Server scripts	Query definition for the query task.

QueryTaskStatus Object Members

The following members are called on [task.QueryTaskStatus](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	QueryTaskStatus.fileId	number (read-only)	Server scripts	Internal ID of the CSV file that query results are exported to.
	QueryTaskStatus.query	query.Query (read-only)	Server scripts	Query definition for the submitted query task.
	QueryTaskStatus.status	string (read-only)	Server scripts	Status of the submitted query task.
	QueryTaskStatus.taskId	string (read-only)	Server scripts	ID of the submitted query task.

RecordActionTask Object Members

The following members are called on [task.RecordActionTaskStatus](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	RecordActionTask.submit()	string	Server scripts	Submits a record action task for processing and returns its task ID.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	RecordActionTask.action	string	Server scripts	The ID of the action to be invoked.
	RecordActionTask.condition	Object	Server scripts	The condition used to select record IDs of records for which the action is to be executed. Only the action.ALL_QUALIFIED_INSTANCES constant is currently supported.
	RecordActionTask.id	string	Server scripts	The ID of the task.
	RecordActionTask.paramCallback()	Object	Server scripts	Function that takes record ID and returns the parameter object for the specified record ID.
	RecordActionTask.params	Object[]	Server scripts	An array of parameter objects. Each object corresponds to one record ID of the record for which the action is to be executed. The object has the following form: {recordId: 1, someParam: 'example1', otherParam: 'example2'}
	RecordActionTask.recordType	string	Server scripts	The record type on which the action is to be performed. For a list of record types, see record.Type .

RecordActionTaskStatus Object Members

The following members are called on [task.RecordActionTaskStatus](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	RecordActionTaskStatus.complete	number (read-only)	Server scripts	The number of record action tasks with a completed status.
	RecordActionTaskStatus.errors	Object (read-only)	Server scripts	The error details of failed action executions. The value of the property is the record instance ID and the corresponding error details. The error details are returned in an unnamed object with two properties: code and message.
	RecordActionTaskStatus.failed	number (read-only)	Server scripts	The number of record action tasks with a failed status.
	RecordActionTaskStatus.pending	number (read-only)	Server scripts	The number of record action tasks with a pending status.
	RecordActionTaskStatus.results	Object (read-only)	Server scripts	The results of successfully executed record action tasks. The value of the property is the task instance ID and the corresponding action result.
	RecordActionTaskStatus.status	string (read-only)	Server scripts	Represents the record action task status. Returns a value from the task.TaskStatus enum.
	RecordActionTaskStatus.succeeded	number (read-only)	Server scripts	The number of record action tasks with a succeeded status.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	RecordActionTask.Status.taskId	string (read-only)	Server scripts	The task ID associated with the specified task.

ScheduledScriptTask Object Members

The following members are called on [task.ScheduledScriptTask](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	ScheduledScriptTask.submit()	string	Server scripts	Directs NetSuite to place a scheduled script deployment into the NetSuite scheduling queue and returns a unique ID for the task.
Property	ScheduledScriptTask.deploymentId	string	Server scripts	Script ID (as a string), for the script deployment record associated with a task.ScheduledScriptTask object.
	ScheduledScriptTask.id	string	Server scripts	The ID of the task.
	ScheduledScriptTask.params	Object	Server scripts	Object with key/value pairs that override the static script parameter field values on the script deployment.
	ScheduledScriptTask.scriptId	number string	Server scripts	Internal ID (as a number), or script ID (as a string) for the script record associated with a task.ScheduledScriptTask object.

ScheduledScriptTaskStatus Object Members

The following members are called on [task.ScheduledScriptTaskStatus](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	ScheduledScriptTaskStatus.deploymentId	string (read-only)	Server scripts	Script ID for a script deployment record associated with a specific task.ScheduledScriptTask object.
	ScheduledScriptTaskStatus.scriptId	number (read-only)	Server scripts	Internal ID for a script record associated with a specific task.ScheduledScriptTask object.
	ScheduledScriptTaskStatus.status	string (read-only)	Server scripts	Status for a scheduled script task. Returns a task.TaskStatus enum value.
	ScheduledScriptTaskStatus.taskId	string (read-only)	Server scripts	The task ID associated with the specified task.

SearchTask Object Members

The following members are called on [task.SearchTask](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	SearchTask.addInboundDependency()	void	Server scripts	Adds a scheduled script task or map/reduce script task to the search task as a dependent script. Dependent scripts are processed automatically when the search task is complete. For more information, see the help topic SuiteCloud Processors .
	SearchTask.submit()	string	Server scripts	Places the asynchronous search initiation task into the SuiteScript task queue, and returns a unique ID for the task.
Property	SearchTask.fileId	number	Server scripts	ID of the CSV file to export search results into.
	SearchTask.filePath	string	Server scripts	Path of the CSV file to export search results into.
	SearchTask.id	string	Server scripts	The ID of the task.
	SearchTask.inboundDependencies	Object[] (read-only)	Server scripts	Key/value pairs to describe the dependent scripts added to the search task.
	SearchTask.savedSearchId	number	Server scripts	ID of the saved search to be executed during the task.

SearchTaskStatus Object Members

The following members are called on [task.SearchTaskStatus](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	SearchTaskStatus.fileId	number (read-only)	Server scripts	ID of the CSV file into which search results are exported.
	SearchTaskStatus.savedSearchId	number (read-only)	Server scripts	ID of the saved search executed during the task.
	SearchTaskStatus.status	string (read-only)	Server scripts	Status of an asynchronous search task placed in the NetSuite task queue. Returns one of the task.TaskStatus enum values.
	SearchTaskStatus.taskId	string (read-only)	Server scripts	ID of the asynchronous task.

SuiteQLTask Object Members

The following members are called on [task.SuiteQLTask](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	SuiteQLTask.addInboundDependency(options)	void	Server scripts	Adds a scheduled script task or map/reduce script task to the

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				SuiteQL task as a dependent task.
	SuiteQLTask.submit()	string	Server scripts	Submits the SuiteQL task for asynchronous processing and returns the task ID.
Property	SuiteQLTask.fileId	number	Server scripts	Internal ID of the CSV file to export SuiteQL query results to. This property is mutually exclusive with the SuiteQLTask.filePath parameter.
	SuiteQLTask.filePath	string	Server scripts	Path of the CSV file to export SuiteQL query results to. This property is mutually exclusive with the SuiteQLTask.fileId property.
	SuiteQLTask.inboundDependencies	Object[]	Server scripts	Key-value pairs that contain information about the dependent tasks added to the SuiteQL task.
	SuiteQLTask.params	Array<string boolean number>	Server scripts	Parameters for the SuiteQL query.
	SuiteQLTask.query	string	Server scripts	SuiteQL query definition for the SuiteQL task.

SuiteQLTaskStatus Object Members

The following members are called on [task.SuiteQLTaskStatus](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	SuiteQLTaskStatus.fileId	number (read-only)	Server scripts	Internal ID of the CSV file that SuiteQL query results are exported to.
	SuiteQLTaskStatus.params	Array<string boolean number> (read-only)	Server scripts	Parameters for the SuiteQL query.
	SuiteQLTaskStatus.query	string (read-only)	Server scripts	SuiteQL query definition for the SuiteQL task.
	SuiteQLTaskStatus.status	string (read-only)	Server scripts	Status of the SuiteQL task.
	SuiteQLTaskStatus.taskId	string (read-only)	Server scripts	ID of the submitted SuiteQL task.

WorkflowTriggerTask Object Members

The following members are called on [task.WorkflowTriggerTask](#).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	WorkflowTriggerTask.submit()	string	Server scripts	Directs NetSuite to place the asynchronous workflow initiation task into the NetSuite scheduling queue and returns a unique ID for the task.
Property	WorkflowTriggerTask.id	string	Server scripts	The ID of the task.
	WorkflowTriggerTask.params	Object	Server scripts	Object that contains key/value pairs to set default values on fields specific to the workflow.
	WorkflowTriggerTask.recordId	number	Server scripts	Internal ID of the workflow definition base record. For example, 55 or 124.
	WorkflowTriggerTask.recordType	string	Server scripts	Record type of the workflow base record. For example, customer, salesorder, or lead.
	WorkflowTriggerTask.workflowId	number string	Server scripts	Internal ID (as a number), or script ID (as a string), for the workflow definition.

WorkflowTriggerTaskStatus Object Members

The following members are called on the [task.WorkflowTriggerTaskStatus](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	WorkflowTriggerTaskStatus.status	string (read-only)	Server scripts	Status for a asynchronous workflow placed in the NetSuite task queue. Returns a value from the task.TaskStatus enum.
	WorkflowTriggerTaskStatus.taskId	string (read-only)	Server scripts	The task ID associated with the specified task.

N/task Module Script Samples

The following script samples demonstrate how to use the features of the N/task module.

Sample 1: Create and submit a map/reduce script

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample submits a map/reduce script task for processing. Before you use this sample, you must create a map/reduce script file, upload the file to your NetSuite account, and create a script record and script deployment record for it. For help working with map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#). You must also edit the sample and replace all hard-coded IDs with values that are valid in your NetSuite account.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/task', 'N/runtime', 'N/email'], function(task, runtime, email) {
6    function submitMapReduceDeployment() {
7
8      // Store the script ID of the script to submit
9      //
10     // Update the following statement so it uses the script ID
11     // of the map/reduce script record you want to submit
12     var mapReduceScriptId = 'customscript_test_mapreduce_script';
13     log.audit('mapreduce id: ', mapReduceScriptId);
14
15     // Create a map/reduce task
16     //
17     // Update the deploymentId parameter to use the script ID of
18     // the deployment record for your map/reduce script
19     var mrTask = task.create({
20       taskType: task.TaskType.MAP_REDUCE,
21       scriptId: mapReduceScriptId,
22       deploymentId: 'customdeploy_test_mapreduce_script'
23     });
24
25     // Submit the map/reduce task
26     var mrTaskId = mrTask.submit();
27
28     // Check the status of the task, and send an email if the
29     // task has a status of FAILED.
30     //
31     // Update the authorId value with the internal ID of the user
32     // who is the email sender. Update the recipientEmail value
33     // with the email address of the recipient.
34     var taskStatus = task.checkStatus(mrTaskId);
35     if (taskStatus.status === 'FAILED') {
36       var authorId = -5;
37       var recipientEmail = 'notify@myCompany.com';
38       email.send({
39         author: authorId,
40         recipients: recipientEmail,
41         subject: 'Failure executing map/reduce job!',
42         body: 'Map reduce task: ' + mapReduceScriptId + ' has failed.'
43       });
44     }
45   }
46
47   submitMapReduceDeployment();
48 });

```

Sample 2: Create and submit an asynchronous search task

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates an asynchronous search task to execute a saved search and export the results of the search into a CSV file stored in the File Cabinet. After the search task is submitted, the sample retrieves the task status using the task ID. Some of the values in this sample are placeholders. Before using this sample, replace all hard-coded values, such as IDs and file paths, with valid values from your NetSuite account. If you run a script with an invalid value, the system may throw an error.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/task'], function(task) {

```

```

6 // Do one of the following:
7 //
8 // - Create a saved search and capture its ID. To do this, you can use
9 // the following code snippet (replacing the id, filters, and columns
10 // values as appropriate):
11 //
12 // var mySearch = search.create({
13 //   type: search.Type.SALES_ORDER,
14 //   id: 'customsearch_my_search',
15 //   filters: [...],
16 //   columns: [...]
17 // });
18 // mySearch.save();
19 // var savedSearchId = mySearch.searchId;
20 //
21 // Use the ID of an existing saved search. This is the approach that
22 // this script sample uses. Update the following statement with the
23 // internal ID of the search you want to use.
24 var savedSearchId = -10;
25
26 // Create the search task
27 var myTask = task.create({
28   taskType: task.TaskType.SEARCH
29 });
30 myTask.savedSearchId = savedSearchId;
31
32 // Specify the ID of the file that search results will be exported into
33 //
34 // Update the following statement so it uses the internal ID of the file
35 // you want to use
36 myTask fileId = 448;
37
38 // Submit the search task
39 var myTaskId = myTask.submit();
40
41 // Retrieve the status of the search task
42 var taskStatus = task.checkStatus({
43   taskId: myTaskId
44 });
45
46 // Optionally, create new variables to represent values used previously in
47 // this script. You may want to use these variables in additional logic you
48 // add to this script.
49 var my fileId = taskStatus fileId;
50 var mySavedSearchId = taskStatus savedSearchId;
51
52 // Optionally, add logic that executes when the task is complete
53 if (taskStatus.status === task.TaskStatus.COMPLETE) {
54   // Add any code that is appropriate. For example, if this script created
55   // a saved search, you may want to delete it.
56 }
57 });

```

Sample 3: Create and submit a task with dependent scripts

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a scheduled script task and a map/reduce script task. It then creates an asynchronous search task and adds the scheduled script task and the map/reduce script task to the search task as dependent scripts. These scripts are processed when the search task is complete. For more information, see the help topic [SuiteCloud Processors](#).

This sample refers to two script parameters: custscript_ss_as_srch_res for the scheduled script, and custscript_mr_as_srch_res for the map/reduce script. These parameters are used to pass the location of the CSV file to the dependent scripts, which is shown in the second and third code samples below. Before

using this sample, create these parameters in the script record. For more information, see the help topic [Creating Script Parameters](#).

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/task'], function(task) {
6     // Specify a file for the search results
7     var asyncSearchResultFile = 'SuiteScripts/ExportFile.csv';
8
9     // Create a scheduled script task
10    var scheduledScript = task.create({
11        taskType: task.TaskType.SCHEDULED_SCRIPT
12    });
13    scheduledScript.scriptId = 'customscript_as_ftr_ss';
14    scheduledScript.deploymentId = 'customdeploy_ss_dpl';
15    scheduledScript.params = {
16        'custscript_ss_as_srch_res' : asyncSearchResultFile
17    };
18
19     // Create a map/reduce script task
20    var mapReduceScript = task.create({
21        taskType: task.TaskType.MAP_REDUCE
22    });
23    mapReduceScript.scriptId = 'customscript_as_ftr_mr';
24    mapReduceScript.deploymentId = 'customdeploy_mr_dpl';
25    mapReduceScript.params = {
26        'custscript_mr_as_srch_res' : asyncSearchResultFile
27    };
28
29     // Create the search task
30    var asyncTask = task.create({
31        taskType: task.TaskType.SEARCH
32    });
33    asyncTask.savedSearchId = 'customsearch35';
34    asyncTask.filePath = asyncSearchResultFile;
35
36     // Add dependent scripts to the search task before it is submitted
37    asyncTask.addInboundDependency(scheduledScript);
38    asyncTask.addInboundDependency(mapReduceScript);
39
40     // Submit the search task
41    var taskId = asyncTask.submit();
42 });

```

To read the contents of the search results file in a dependent scheduled script, consider the following script sample:

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType ScheduledScript
4 */
5 define(['N/file', 'N/log', 'N/email', 'N/runtime'], function(file, log, email, runtime) {
6     // Load the search results file and send an email with the file attached and
7     // the number of rows in the file
8
9     function execute(context) {
10        // Read a CSV file and return the number of rows minus the header row
11        function numberOfRows(csv fileId) {
12            var invoiceFile = file.load({
13                id: csv fileId
14            });
15            var iterator = invoiceFile.lines.iterator();
16            var noOfLines = 0;
17
18            // Skip the first row (the header row)
19            iterator.each(function() {
20                return false;
21            });

```

```

23     // Process the rest of the rows
24     iterator.each(function() {
25         noOfLines++;
26         return true;
27     });
28
29     return noOfLines;
30 }
31
32 // Send an email to the user who ran the script, and attach the
33 // CSV file with the search results
34 function sendEmailWithAttachment(csv fileId) {
35     var noOfRows = numberOfRows(csv fileId);
36     var userId = runtime.getCurrentUser().id;
37     var fileObj = file.load({
38         id: csv fileId
39     });
40
41     email.send({
42         author: userId,
43         recipients: userId,
44         subject: 'Search completed',
45         body: 'CSV file attached, ' + noOfRows + ' record(s) found.',
46         attachments: [fileObj]
47     });
48 }
49
50 // Retrieve the ID of the search results file
51 //
52 // Update the name parameter to use the script ID of the original
53 // search task
54 var res fileId = runtime.getCurrentScript().getParameter({
55     name: 'custscript_ss_as_srch_res'
56 });
57
58 if (!res fileId) {
59     log.error('Could not obtain file content from the specified ID.');
60     return;
61 }
62
63 log.debug({
64     title: 'search - numberOfRows',
65     details: numberOfRows(res fileId)
66 });
67 sendEmailWithAttachment(res fileId);
68 }
69
70 return {
71     execute: execute
72 };
73 });

```

To read the contents of the search results file in a dependent map/reduce script, consider the following script sample:

```

1 /**
2 * @NApiVersion 2.x
3 * @NScriptType MapReduceScript
4 * @NModuleScope SameAccount
5 */
6 define(['N/runtime', 'N/file', 'N/log', 'N/email'], function(runtime, file, log, email) {
7     // Load the search results file, count the number of letters in the file, and
8     // store this count in another file
9
10    function getInputData() {
11        // Retrieve the ID of the search results file
12        //
13        // Update the completionScriptParameterName value to use the script
14        // ID of the original search task
15        var completionScriptParameterName = 'custscript_mr_as_srch_res';
16        var res fileId = runtime.getCurrentScript().getParameter({
17            name: completionScriptParameterName

```

```

18     });
19
20     if (!res fileId) {
21         log.error({
22             details: 'res fileId is not valid. Please check the script parameter stored in the completionScriptParameterName
variable in getInputData().'
23         });
24     }
25
26     return {
27         type: 'file',
28         id: res fileId
29     };
30 }
31
32 function map(context) {
33     var email = context.value.split(',') [1];
34     if ("Email" !== email) {
35         var splitEmail = email.split('@');
36         context.write(splitEmail[splitEmail.length-1], 1);
37     }
38 }
39
40 function reduce(context) {
41     context.write(context.key, context.values.length);
42 }
43
44 function summarize(summary) {
45     var type = summary.toString();
46     log.audit({title: type + ' Usage Consumed ', details: summary.usage});
47     log.audit({title: type + ' Concurrency Number ', details: summary.concurrency});
48     log.audit({title: type + ' Number of Yields ', details: summary.yields});
49
50     var contents = '';
51     summary.output.iterator().each(function(key, value) {
52         contents += (key + ' ' + value + '\n');
53         return true;
54     });
55
56     // Create the output file
57     //
58     // Update the name parameter to use the file name of the output file
59     var fileObj = file.create({
60         name: 'domainCount.txt',
61         fileType: file.Type.PLAINTEXT,
62         contents: contents
63     );
64
65     // Specify the folder location of the output file, and save the file
66     //
67     // Update the fileObj.folder property with the ID of the folder in
68     // the file cabinet that contains the output file
69     fileObj.folder = -15;
70     fileObj.save();
71 }
72
73 return {
74     getInputData: getInputData,
75     map: map,
76     reduce: reduce,
77     summarize: summarize
78 };
79 });

```

Sample 4: Submit a task and check its status

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to submit a record action task and then check its status. For details about record action tasks, see [task.RecordActionTask](#) and [task.RecordActionTaskStatus](#).

```

1  /**
2  * @NApiVersion 2.x
3  */
4
5  require(['N/task'], function(task) {
6      var recordActionTask = task.create({
7          taskType: task.TaskType.RECORD_ACTION
8      });
9      recordActionTask.recordType = 'timebill';
10     recordActionTask.action = 'approve';
11     recordActionTask.params = [
12         {recordId: 1, note: 'This is a note for 1'},
13         {recordId: 5, note: 'This is a note for 5'},
14         {recordId: 23, note: 'This is a note for 23'}
15     ];
16
17     var handle = recordActionTask.submit();
18
19     var res = task.checkStatus({
20         taskId: handle
21     }); // Returns a RecordActionTaskStatus object
22     log.debug('Initial status: ' + res.status);
23 });

```

task.CsvImportTask

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>The properties of a CSV import task. Use the methods and properties for this object to submit a CSV import task into the task queue and asynchronously import record data into NetSuite.</p> <p>Use the CsvImportTask Object to perform the following types of tasks:</p> <ul style="list-style-type: none"> ■ Automate standard record data import for SuiteApp installations, demo environments, and testing environments. ■ Import data on a schedule using a scheduled script. ■ Build integrated CSV imports with RESTlets. <p>Use the following process to import CSV data with CsvImportTask:</p> <ul style="list-style-type: none"> ■ In the NetSuite UI, run the Import Assistant to set up the CSV mapping and import options. You must run the Import Assistant to set up the necessary mapping for the CSV import. You can use a sample file or files to set up the mapping. Note the following information: <ul style="list-style-type: none"> □ Script ID for import map. □ Any required linked files. For more information, see the help topic Importing CSV Files with the Import Assistant. ■ Use <code>task.create(options)</code> to create the CsvImportTask object.
---------------------------	---

	<ul style="list-style-type: none"> ■ Use the CsvImportTask object properties to set the script and deployment properties. ■ Use CsvImportTask.submit() to submit the import task to the NetSuite task queue. ■ Use the properties for the task.CsvImportTaskStatus object to get the status of the import process. <p>Use the following guidelines with the CsvImportTask Object:</p> <ul style="list-style-type: none"> ■ CSV imports performed within scripts are subject to the existing application limit of 25,000 records. ■ You cannot import data that is imported by (2-step) assistants in the UI, because these import types do not support saved import maps. This limitation applies to budget, single journal entry, single inventory worksheet, project tasks, and website redirects imports. ■ This object has access only to the field mappings of a saved import map; it does not have access to advanced import options defined in the Import Assistant, such as multi-threading and multiple queues. <p>Even if you set options to use multiple threads or queues for an import job and then save the import map, these settings are not available to CsvImportTask. When this object submits a CSV import job based on the saved import map, a single thread and single queue are used.</p> <p>For a complete list of this object's methods and properties, see CsvImportTask Object Members.</p>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var scriptTask = task.create({taskType: task.TaskType.CSV_IMPORT});
4 scriptTask.mappingId = 51;
5 var f = file.load('SuiteScripts/custjoblist.csv');
6 scriptTask.importFile = f;
7 scriptTask.linkedFiles = {'addressbook': 'street,city\nval1,val2', 'purchases': file.load('SuiteScripts/other.csv')};
8 var csvImportTaskId = scriptTask.submit();
9 ...
10 //Add additional code

```

CsvImportTask.submit()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Directs NetSuite to place a CSV import task into the NetSuite task queue and returns a unique ID for the task. Use CsvImportTaskStatus.status to view the status of a submitted task.</p> <p>This method throws errors resulting from inline validation of CSV file data before the import of data begins (the same validation that is performed between the mapping step and the save step in the Import Assistant). Any errors that occur during the import job are recorded in the CSV response file, as they are for imports initiated through the Import Assistant.</p>
---------------------------	---

Returns	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	100 units
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
FAILED_TO_SUBMIT_JOB_REQUEST_1	Failed to submit job request: {reason}	Task cannot be submitted.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var csvImportTaskId = csvTask.submit();
4 ...
5 //Add additional code

```

CsvImportTask.id

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the task.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.CsvImportTask
Sibling Object Members	CsvImportTask Object Members
Since	

CsvImportTask.importFile

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	CSV file to import. Use a file.File object or a string that represents the CSV text to be imported.
-----------------------------	---

Type	<code>file.File</code> string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var f = file.load('SuiteScripts/custjoblist.csv');
4 scriptTask.importFile = f;
5 ...
6 //Add additional code

```

CsvImportTask.linkedFiles



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	A map of key/value pairs that sets the data to be imported in a linked file for a multi-file import job, by referencing a file in the File Cabinet or the raw CSV data to import. The key is the internal ID of the record sublist for which data is being imported and the value is either a <code>file.File</code> object or the raw CSV data to import. You can assign multiple types of values to the linkedFiles property.
Type	Object
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 scriptTask.linkedFiles = {'addressbook': 'street,city\nval1,val2', 'purchases': file.load('SuiteScripts/other.csv')};
4 ...
5 //Add additional code

```

CsvImportTask.mappingId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Script ID or internal ID of the saved import map that you created when you ran the Import Assistant. See task.CsvImportTask .
Type	number string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 scriptTask.mappingId = 51;
4 ...
5 //Add additional code

```

CsvImportTask.name

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Name for the CSV import task. You can optionally set a different name for a scripted import task. In the UI, this name appears on the CSV Import Job Status page.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 csvTask.name = 'Import Entities'

```

```

4 | ...
5 | //Add additional code

```

CsvImportTask.queueId

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Overrides the Queue Number property under Advanced Options on the Import Options page of the Import Assistant. Use this property to programmatically select an import queue and improve performance during the import.
	<p>i Note: This property is only available if you have a SuiteCloud Plus license. For more information about using multiple queues when importing CSV files, see the help topics Queue Number and Use Multiple Threads and Multiple Queues to Run CSV Import Jobs.</p>
Type	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 | //Add additional code
2 | ...
3 | scriptTask.queueId = 2;
4 | ...
5 | //Add additional code

```

task.CsvImportTaskStatus

i Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The status of a CSV import task placed into the NetSuite scheduling queue. Use task.checkStatus(options) with the unique ID for the CSV import task to get the CsvImportTaskStatus object. For a complete list of this object's properties, see CsvImportTaskStatus Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module

Since	2015.2
-------	--------

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var csvTaskStatus = task.checkStatus({
4     taskId: csvTaskId
5 });
6 if (csvTaskStatus.status === task.TaskStatus.FAILED)
7 ...
8 //Add additional code

```

CsvImportTaskStatus.status



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status for a CSV import task. Returns a task.TaskStatus enum value.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus({
4     taskId: scriptTaskId
5 });
6 log.audit({
7     title: 'Status',
8     details: summary.status
9 });
10 ...
11 //Add additional code

```

CsvImportTaskStatus.taskId

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Property Description	The task ID associated with the specified task.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.CsvImportTaskStatus
Sibling Object Members	CsvImportTaskStatus Object Members
Since	

Errors

Error Code	Error Message	Thrown If
READ_ONLY		Setting the property is attempted.

task.EntityDeduplicationTask

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Object Description	<p>All the properties of a merge duplicate records task request. Use the methods and properties of this object to submit a merge duplicate record job task into the NetSuite task queue.</p> <p>When you submit a merge duplicate record task to NetSuite, SuiteScript enables you to use all of the same functionality available through the UI. Use SuiteScript to use the predefined duplicate detection rules, or you can define your own. After the records are merged or deleted, in the UI, the records no longer appear as duplicates at Lists > Mass Update > Entity Duplicate Resolution .</p> <p>For more information about merging duplicate records in NetSuite, see the help topic Merging or Deleting Duplicate Records.</p> <p>To use the EntityDeduplicationTask Object:</p> <ul style="list-style-type: none"> ■ Use task.create(options) to create the EntityDeduplicationTask object. ■ Use EntityDeduplicationTask.entityType to select the entity type on which you want to merge duplicate records. ■ Use EntityDeduplicationTask.dedupeMode to select the action to take for the duplicate records. ■ Use a EntityDeduplicationTask.masterSelectionMode enum value to identify which record to use as the master record in the merge. ■ If you use MasterSelectionMode.SELECT_BY_ID for the master selection mode, set the ID of the master record with EntityDeduplicationTask.masterRecordId. ■ Identify the duplicate records. Use the search.duplicates(options) method in the N/search Module to find the duplicate records.
---------------------------	---

	<ul style="list-style-type: none"> ■ Use EntityDeduplicationTask.submit() to submit the merge duplicate record task to the NetSuite task queue. ■ Use the properties for the task.EntityDeduplicationTaskStatus object to get the status of the merge duplicate record task. <p>Use the following guidelines with the EntityDeduplicationTask Object:</p> <ul style="list-style-type: none"> ■ You can only submit 200 records in a single merge duplicate records task. ■ The merge duplicate functionality on non-entity records is not supported in SuiteScript. ■ You must have full access to the Duplicate Record Management permission to merge duplicates. <p>For a complete list of this object's methods and properties, see EntityDeduplicationTask Object Members.</p>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var dedupeTask = task.create({taskType: task.TaskType.ENTITY_DEDUPLICATION});
4 dedupeTask.entityType = task.DedupeEntityType.CUSTOMER;
5 dedupeTask.dedupeMode = task.DedupeMode.MERGE;
6 dedupeTask.masterSelectionMode = task.MasterSelectionMode.MOST_RECENT_ACTIVITY;
7 dedupeTask.recordIds = ['107', '110'];
8 var dedupeTaskId = dedupeTask.submit();
9 ...
10 //Add additional code

```

EntityDeduplicationTask.submit()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Directs NetSuite to place the merge duplicate records task into the NetSuite task queue and returns a unique ID for the task. Use EntityDeduplicationTaskStatus.status to view the status of a submitted task.
Returns	string The task ID
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	100 units
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
FAILED_TO_SUBMIT_JOB_REQUEST_1	Failed to submit job request: {reason}	Task cannot be submitted.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var dedupeTaskId = dedupeTask.submit();
4 ...
5 //Add additional code

```

EntityDeduplicationTask.dedupeMode

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The mode in which to merge or delete duplicate records. Use a task.DedupeMode enum value to set this property.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 dedupeTask.dedupeMode = task.DedupeMode.MERGE;
4 ...
5 //Add additional code

```

EntityDeduplicationTask.entityType

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The type of entity on which you want to merge duplicate records.
-----------------------------	--

	<p>Use a task.DedupeEntityType enum value to set the value.</p> <p>Note: If you set entityType to CUSTOMER, the system will automatically include prospects and leads in the task request.</p>
Type	string
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 dedupeTask.entityType = task.DedupeEntityType.CUSTOMER;
4 ...
5 //Add additional code

```

EntityDeduplicationTask.id

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the task.
Type	string
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/task Module
Parent Object	task.EntityDeduplicationTask
Sibling Object Members	EntityDeduplicationTask Object Members
Since	

EntityDeduplicationTask.masterRecordId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>When you merge duplicate records, you can delete all duplicates for a record or merge information from the duplicate records into the master record.</p> <p>Use this property to set the ID of the master record that you want to use as the master record in the merge.</p>
-----------------------------	---

	 Important: You must also select SELECT_BY_ID for the <code>EntityDeduplicationTask.masterSelectionMode</code> property, or NetSuite ignores this setting.
Type	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 dedupeTask.masterSelectionMode = task.MasterSelectionMode.SELECT_BY_ID;
4 dedupeTask.masterRecordId = 107;
5 ...
6 //Add additional code

```

EntityDeduplicationTask.masterSelectionMode

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	When you merge duplicate records, you can delete all duplicates for a record or merge information from the duplicate records into the master record. Set this property to determine which of the duplicate records to keep or select the master record to use by ID. Use the <code>task.MasterSelectionMode</code> enum to set the value.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 dedupeTask.masterSelectionMode = task.MasterSelectionMode.MOST_RECENT_ACTIVITY;
4 ...
5 //Add additional code

```

EntityDeduplicationTask.recordIds

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Number array of record internal IDs to perform the merge or delete operation on. You can use the search.duplicates(options) method to identify duplicate records or create an array with record internal IDs.
Type	number[]
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 dedupeTask.recordIds = ['107', '110'];
4 ...
5 //Add additional code

```

task.EntityDeduplicationTaskStatus

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	The status of a merge duplicate record task placed into the NetSuite task queue by EntityDeduplicationTask.submit() . Use task.checkStatus(options) with the unique ID for the merge duplicate records task to get this Object. For a complete list of this object's properties, see EntityDeduplicationTaskStatus Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```
1 //Add additional code
```

```

2 ...
3 var dedupeTaskStatus = task.checkStatus({
4   taskId: taskId
5 });
6 if (dedupeTaskStatus.status === task.TaskStatus.FAILED)
7 ...
8 //Add additional code

```

EntityDeduplicationTaskStatus.status

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status for a merge duplicate record task. Returns a task.TaskStatus enum value.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus({
4   taskId: scriptTaskId
5 });
6 log.audit({
7   title: 'Status',
8   details: summary.status
9 });
10 ...
11 //Add additional code

```

EntityDeduplicationTaskStatus.taskId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The task ID associated with the specified task.
Type	string (read-only)
Supported Script Types	Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.EntityDeduplicationTaskStatus
Sibling Object Members	EntityDeduplicationTaskStatus Object Members
Since	

Errors

Error Code	Error Message	Thrown If
READ_ONLY		Setting the property is attempted.

task.MapReduceScriptTask



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The properties of a map/reduce script deployment. You can use this object to programmatically submit a script deployment for processing. To use the MapReduceScriptTask object: <ul style="list-style-type: none">■ In the NetSuite UI, create the script record and script deployment records.■ Use task.create(options) to create the MapReduceScriptTask object.■ Use the MapReduceScriptTask object properties to set the script and deployment properties.■ Use MapReduceScriptTask.submit() to submit the deployment for processing.■ Use the properties for the task.MapReduceScriptTaskStatus object to get the status of the map/reduce script. For a complete list of this object's methods and properties, see MapReduceScriptTask Object Members . For general information about map/reduce scripts, see the help topic Map/Reduce Key Concepts .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var mrTask = task.create({taskType: task.TaskType.MAP_REDUCE});
4 mrTask.scriptId = mapReducescriptId;

```

```

5 mrTask.deploymentId = custdeploy1;
6 var mrTaskId = mrTask.submit();
7 ...
8 //Add additional code

```



Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTask.submit()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Submits a map/reduce script deployment for processing.</p> <p>For more information, see task.MapReduceScriptTask.</p> <p>Additionally, note that a map/reduce script can be submitted for processing only if there is no unfinished map/reduce script task for the same script ID and script deployment ID. For this reason, if a map/reduce script resubmits itself, the actual resubmit does not occur until the current execution completes. This delay is necessary to avoid the existence of two unfinished tasks for the same deployment of the same script. Therefore, if a map/reduce script uses the submit() method to resubmit itself, then at runtime, no task ID is returned when the map/reduce script is submitted.</p> <p>Executing a map/reduce script from a server script requires either the Administrator role or a role with the SuiteScript Scheduling permission.</p> <p>For general information about the execution of map/reduce scripts, see the help topic SuiteScript 2.0 Map/Reduce Script Submission.</p>
Returns	string The task ID, except as noted above.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
FAILED_TO_SUBMIT_JOB_REQUEST_1	Failed to submit job request: {reason}	Task cannot be submitted.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var mrTaskId = mrTask.submit();
4 ...

```

```
5 //Add additional code
```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTask.deploymentId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Script ID (as a string), for the script deployment record for a map/reduce script.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```
1 //Add additional code
2 ...
3 mrTask.deploymentId = custdeploy1;
4 ...
5 //Add additional code
```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTask.id

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the task.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.MapReduceScriptTask
Sibling Object Members	MapReduceScriptTask Object Members
Since	

MapReduceScriptTask.params



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The key/value pairs that override static script parameter field values on the script deployment record. Use these parameters on a <code>task.MapReduceScriptTask</code> object to programmatically pass values to the script deployment. For more information about script parameters, see the help topic Creating Script Parameters Overview .
Type	Object
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3
4 mrTask.params = {doSomething: true};
5 ...
6 //Add additional code

```



Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTask.scriptId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID (as a number), or script ID (as a string), for the map/reduce script record.
Type	number string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```
1 //Add additional code
```

```

2 ...
3 var mapReducescriptId = 34;
4 ...
5 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

task.MapReduceScriptTaskStatus

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The status of a map/reduce script deployment that was submitted for processing. Use <code>task.checkStatus(options)</code> with the unique ID for the map/reduce script task to get the <code>MapReduceScriptTaskStatus</code> object. For a complete list of this object's methods and properties, see MapReduceScriptTaskStatus Object Members . For general information about the execution of map/reduce scripts, see the help topic SuiteScript 2.0 Map/Reduce Script Submission .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 if (summary.stage === task.MapReduceStage.SUMMARIZE)
5   log.audit('Almost done...');
6 ...
7 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getCurrentTotalSize()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total size in bytes of all stored work in progress by a <code>task.MapReduceScriptTask</code> .
Returns	number
Supported Script Types	Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	25 units
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 var total = summary.getCurrentTotalSize()
5 log.audit({
6   title: 'Size of Remaining Data to Process',
7   details: total
8 });
9 ...
10 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getPendingMapCount()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total number of records or rows not yet processed by the map stage of a task.MapReduceScriptTask . Use the MapReduceScriptTaskStatus.stage property to get the current stage. For general information about map/reduce stages, see the help topics Map/Reduce Key Concepts and SuiteScript 2.0 Map/Reduce Script Stages .
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = taskStatus.getPendingMapCount();
4 log.audit('Pending Map Count: ' + summary);

```

```

5 ...
6 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getPendingMapSize()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total number of bytes not yet processed by the map stage, as a component of total size, of a <code>task.MapReduceScriptTask</code> . Use the MapReduceScriptTaskStatus.stage property to get the current stage. For general information about map/reduce stages, see the help topics Map/Reduce Key Concepts and SuiteScript 2.0 Map/Reduce Script Stages .
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	25 units
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = taskStatus.getPendingMapSize();
4 log.audit('Pending Map Size: ' + summary);
5 ...
6 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getPendingOutputCount()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total number of records or rows not yet processed by a <code>task.MapReduceScriptTask</code> .
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	10 units
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 var total = summary.getPendingOutputCount()
5 log.audit({
6   title: 'Count',
7   details: total
8 });
9 ...
10 //Add additional code

```

 **Note:** For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getPendingOutputSize()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total size in bytes of all key/value pairs written as output, as a component of total size, by a task.MapReduceScriptTask .
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	25 units
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 var total = summary.getPendingOutputSize()
5 log.audit({
6   title: 'Size',
7   details: total
8 });
9 ...
10 //Add additional code

```



Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getPendingReduceCount()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total number of records or rows not yet processed by the reduce stage of a task.MapReduceScriptTask . Use the MapReduceScriptTaskStatus.stage property to get the current stage. For general information about the reduce stage and other map/reduce stages, see the help topics Map/Reduce Key Concepts and SuiteScript 2.0 Map/Reduce Script Stages .
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = taskStatus.getPendingReduceCount();
4 log.audit({
5   details: 'Pending Reduce Count: ' + summary
6 });
7 ...
8 //Add additional code

```



Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getPendingReduceSize()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total number of bytes not yet processed by the reduce stage, as a component of total size, of a task.MapReduceScriptTask . Use the MapReduceScriptTaskStatus.stage property to get the current stage.
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	25 units
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var summary = taskStatus.getPendingReduceSize();
4 log.audit({
5   details: 'Pending Reduce Size: ' + summary
6 });
7 ...
8 //Add additional code

```



Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getPercentageCompleted()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the current percentage complete for the current stage of a task.MapReduceScriptTask . Use the MapReduceScriptTaskStatus.stage property to get the current stage. For general information about map/reduce stages, see the help topics Map/Reduce Key Concepts and SuiteScript 2.0 Map/Reduce Script Stages .
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var completion = taskStatus.getPercentageCompleted();

```

```

4 log.audit('Percentage Completed: ' + completion);
5 ...
6 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getTotalMapCount()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total number of records or rows passed as input to the map stage of a task.MapReduceScriptTask . Use the MapReduceScriptTaskStatus.stage property to get the current stage. For general information about map/reduce stages, see the help topics Map/Reduce Key Concepts and SuiteScript 2.0 Map/Reduce Script Stages .
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = taskStatus.getTotalMapCount();
4 log.audit('Total Map Count: ' + summary);
5 ...
6 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getTotalOutputCount()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total number of key/value pairs passed as inputs to the SUMMARIZE phase of a task.MapReduceScriptTask . Use the MapReduceScriptTaskStatus.stage property to get the current stage.
Returns	number
Supported Script Types	Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 var total = summary.getTotalOutputCount()
5 log.audit({
6   title: 'Total Entries Passed to Output',
7   details: total
8 });
9 ...
10 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.getTotalReduceCount()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the total number of record or row inputs to the reduce stage of a task.MapReduceScriptTask . Use the MapReduceScriptTaskStatus.stage property to get the current stage. For general information about map/reduce stages, see the help topics Map/Reduce Key Concepts and SuiteScript 2.0 Map/Reduce Script Stages .
Returns	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	10 units
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = taskStatus.getTotalReduceCount();
4 log.audit({

```

```

5   details: 'Reduce Count: ' + summary
6 });
7 ...
8 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.deploymentId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Script ID for a script deployment record associated with a specific task.MapReduceScriptTask .
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 log.audit({
5   title: 'Deployment ID',
6   details: summary.deploymentId
7 });
8 ...
9 ...
10 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.scriptId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID for a map/reduce script record associated with a specific task.MapReduceScriptTask .
-----------------------------	--

Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 log.audit({
5   title: 'Script ID',
6   details: summary.scriptId
7 });
8 ...
9 ...
10 //Add additional code

```



Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.stage



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Current stage of processing for a map/reduce script deployment instance. Returns a task.MapReduceStage enum value. For general information about map/reduce stages, see the help topic Map/Reduce Key Concepts . For information about the execution of map/reduce scripts, see the help topic SuiteScript 2.0 Map/Reduce Script Submission .
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 if (summary.stage === task.MapReduceStage.SUMMARIZE)
5   log.audit({
6     details: 'Almost done...'
7   });
8 ...
9 //Add additional code

```



Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.status



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status of a map/reduce script deployment that was submitted for processing. Returns a task.TaskStatus enum value. For general details about the execution of map/reduce scripts, see the help topic SuiteScript 2.0 Map/Reduce Script Submission .
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 log.audit({
5   title: 'Status',
6   details: summary.status
7 });

```

```

8
9 ...
10 //Add additional code

```

Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

MapReduceScriptTaskStatus.taskId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The task ID associated with the specified task.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.MapReduceScriptTaskStatus
Sibling Object Members	MapReduceScriptTaskStatus Object Members
Since	

Errors

Error Code	Error Message	Thrown If
READ_ONLY		Setting the property is attempted.

task.QueryTask

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	An asynchronous query task. Use the methods and properties for this object to submit a query task to the NetSuite task queue and execute it asynchronously. You can create a task.QueryTask object using task.create(options) . You can populate the properties of this object as you create it. With a task.QueryTask object, you can do the following: <ul style="list-style-type: none">■ Specify the query to submit using the QueryTask.query property. This property accepts a query.Query object that was constructed using the N/query module. For more information, see N/query Module.■ Set the file ID or file path of a CSV file in the File Cabinet. Query results are exported to this file. Use the QueryTask fileId property to specify a file ID, or use QueryTask.filePath to specify a file path. Only one of these properties can be set at the same time. If both are set, an error occurs.■ Add dependent tasks to the query task using QueryTask.addInboundDependency(options). Dependent tasks are processed automatically when the query task is complete.■ Submit the query task to the task queue using QueryTask.submit().
---------------------------	--

	<ul style="list-style-type: none"> Get the status of a query task using the properties of an associated <code>task.QueryTaskStatus</code> object. The <code>task.checkStatus(options)</code> method returns a <code>task.QueryTaskStatus</code> object when you specify a task ID that represents a query task.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Methods and Properties	QueryTask Object Members
Since	2020.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var myQueryTask = task.create({
4   taskType: task.TaskType.QUERY
5 });
6 ...
7 // Add additional code

```

QueryTask.addInboundDependency(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Adds a scheduled script task or map/reduce script task as a dependent task to the query task. Dependent tasks are processed automatically using SuiteCloud Processors when the query task is complete. For more information, see the help topic SuiteCloud Processors.</p> <p>You can add a dependent task in the following ways:</p> <ul style="list-style-type: none"> Provide a <code>task.ScheduledScriptTask</code> object or <code>task.MapReduceScriptTask</code> object that represents the dependent task as the only parameter to this method. Provide an Object as the only parameter that includes values for the following properties: <ul style="list-style-type: none"> taskType scriptId deploymentId (optional) params (optional) <p>For descriptions of these properties, see Parameters.</p> <p>Important: You can add only scheduled scripts or map/reduce scripts as dependent tasks to asynchronous query tasks. Other script types are not supported. You can add only one dependent task per call to <code>QueryTask.addInboundDependency(options)</code>.</p> <p>When you use this method to add a dependent task, the added task is considered an inbound dependency of the query task. For example, if you add a scheduled script task to a query task as a dependent task, the scheduled script depends on the query task.</p>
---------------------------	---

	Because <code>QueryTask.addInboundDependency(options)</code> is called on the query task, any dependent task that you add is considered an inbound dependency.
Returns	<code>void</code>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/task Module
Parent Object	task.QueryTask
Sibling Object Members	QueryTask Object Members
Since	2020.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.scriptId</code>	<code>string</code>	required	The script ID of the scheduled script record or map/reduce script record for the dependent task.
<code>options.taskType</code>	<code>string</code>	required	<p>The type of dependent task.</p> <p>This property uses one of the following values in the task.TaskType enum:</p> <ul style="list-style-type: none"> ■ <code>task.TaskType.SCHEDULED_SCRIPT</code> ■ <code>task.TaskType.MAP_REDUCE</code>
<code>options.deploymentId</code>	<code>string</code>	optional	<p>The script ID of the script deployment record for the dependent task.</p> <p>To use this property, the scheduled script or map/reduce script for the dependent task must be deployed. If you do not specify a value for this property, an available deployment is selected automatically.</p>
<code>options.params</code>	<code>Object</code>	optional	<p>The parameters for the scheduled script or map/reduce script.</p> <p>To use this property, you must first create script parameters that are required for the dependent task. For example, if your dependent task requires a reference to the CSV result file, you must create a script parameter that maps the CSV result file ID to the file name, as in the following example:</p> <pre>1 {</pre>

Parameter	Type	Required / Optional	Description
			<pre> 2 'custscript_query_res' : 'result 3 File.csv' 3 </pre> <p>For more information about script parameters, see the help topic Creating Script Parameters (Custom Fields).</p>

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // Make sure to use a script ID that represents a scheduled script in your account
4 var myScheduledScript = task.create({
5     taskType: task.TaskType.SCHEDULED_SCRIPT,
6    scriptId: 'customscript_as_ftr_ss'
7 });
8
9 // myQueryTask is an existing task.QueryTask object
10 myQueryTask.addInboundDependency(myScheduledScript);
11 ...
12 // Add additional code

```

QueryTask.submit()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Submits the query task for asynchronous processing. This method returns a unique ID for the query task. When the submission is successful, this method adds the script IDs of any dependent tasks (added using QueryTask.addInboundDependency(options)) to the QueryTask.inboundDependencies property. Use the task.QueryTaskStatus object to view the status of a submitted query task. The task.checkStatus(options) method returns a task.QueryTaskStatus object when you specify a task ID that represents a query task.
Returns	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	100 units
Module	N/task Module
Parent Object	task.QueryTask
Sibling Object Members	QueryTask Object Members
Since	2020.2

Errors

Error Code	Thrown If
ASYNC_QUERY_DEPENDENCY_MR_ALREADY_SUBMITTED	A dependent map/reduce script task is already submitted and is not complete.
ASYNC_QUERY_DEPENDENCY_MR_INCORRECT_STATUS	The status of the script deployment record for the specified dependent map/reduce script task has a value other than "Not Scheduled".
ASYNC_QUERY_DEPENDENCY_SS_ALREADY_SUBMITTED	A dependent scheduled script task is already submitted and is not complete.
ASYNC_QUERY_DEPENDENCY_SS_INCORRECT_STATUS	The status of the script deployment record for the specified dependent scheduled script task has a value other than "Not Scheduled".
ASYNC_QUERY_DEPLOYMENT_FOR_DEPENDENCY	A script deployment record for the specified dependent script task is not available for one of the following reasons: <ul style="list-style-type: none"> ■ A script deployment record was not specified when the dependent task was created, and automatic lookup for an available script deployment record failed. ■ The script deployment record specified when the dependent task was created is not found.
ASYNC_QUERY_MULTIPLE_DEPENDENCIES	The same dependent task is passed to this method more than once.
ASYNC_QUERY_QUERY_ID_NOT_FOUND	A query task with the specified script ID is not found.
ASYNC_QUERY_SCRIPT_ID_NOT_FOUND	The specified dependent task is not found.
CANNOT_RESUBMIT_SUBMITTED_ASYNC_QUERY_TASK	The query task was already submitted and completed successfully.
FAILED_TO_SUBMIT_JOB_REQUEST_1	The query task cannot be submitted due to an unexpected error.
MUST_IDENTIFY_A_FILE	The <code>QueryTask.filePath</code> property specifies a folder and not a file.
SSS_MISSING_REQD_ARGUMENT	A required property is not specified.
THAT_RECORD_DOES_NOT_EXIST	The <code>QueryTask.fileId</code> property or <code>QueryTask.filePath</code> property references a file that does not exist.
YOU_DO_NOT_HAVE_ACCESS_TO_THE_MEDIA_ITEM_YOU_SELECTED	You do not have permission to access the file specified by the <code>QueryTask.fileId</code> property or <code>QueryTask.filePath</code> property.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...

```

```

3 | var myQueryTask = task.create({
4 |   taskType: task.TaskType.QUERY
5 | });
6 |
7 | // Set the properties of the myQueryTask object
8 |
9 | // Submit the task
10 | var myTaskId = myQueryTask.submit();
11 |
12 | // Check the task status
13 | var myStatus = task.checkStatus({
14 |   taskId: myTaskId
15 | });
16 | ...
17 | // Add additional code

```

QueryTask.fileId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID of the CSV file to export query results to. The CSV file must be located in the File Cabinet. You can provide a value for this property or the QueryTask.filePath property. Only one of these properties can be set at the same time. If both are set, an error occurs.
Type	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.QueryTask
Sibling Object Members	QueryTask Object Members
Since	2020.2

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You specified values for both QueryTask.fileId and QueryTask.filePath .

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 | // Add additional code
2 | ...
3 | var myQueryTask = task.create({
4 |   taskType: task.TaskType.QUERY
5 | });
6 |
7 | myQueryTask.fileId = 18;
8 |
9 | // Add additional code

```

QueryTask.filePath

Note: The content in this help topic pertains to SuiteScript 2.0.	
Property Description	Path of the CSV file to export query results to. The CSV file must be located in the File Cabinet. You can provide a value for this property or the QueryTask.fileId property. Only one of these properties can be set at the same time. If both are set, an error occurs.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.QueryTask
Sibling Object Members	QueryTask Object Members
Since	2020.2

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You specified values for both QueryTask.fileId and QueryTask.filePath .

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var myQueryTask = task.create({
4   taskType: task.TaskType.QUERY
5 });
6
7 myQueryTask.filePath = 'ExportFolder/export.csv';
8 ...
9 // Add additional code

```

QueryTask.inboundDependencies

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Key-value pairs that contain information about the dependent tasks added to the query task. Use this property to verify the properties of dependent tasks after you add the tasks using QueryTask.addInboundDependency(options) . This property uses nested objects to store information about each dependent task. A nested object is included for each dependent task added to the query task, and these objects are referenced by their index (starting at 0). Dependent tasks are indexed in the order they are
-----------------------------	---

added to the query task. Each nested object contains the task type, script ID, script deployment ID, and script parameters.

For example, consider a situation in which you add a scheduled script task and a map/reduce script task to a query task as dependent tasks. After you add the dependent tasks but before you submit the query task using `QueryTask.submit()`, the value of the `QueryTask.inboundDependencies` property is similar to the following:

```

1  {"0":
2    {"type":"task.ScheduledScriptTask", "scriptId":"customscript_as_ftr_ss", "deploymentId":"customdeploy_ss_dpl",
3     "params":
4       {"custscript_ss_as_src_res":"SuiteScripts/ExportFile.csv"}
5   },
6  {"1":
7    {"type":"task.MapReduceScriptTask", "scriptId":"customscript_as_ftr_mr", "deploymentId":"customdeploy_mr_dpl",
8     "params":
9       {"custscript_mr_as_src_res":"SuiteScripts/ExportFile.csv"}
10  }
11 }
```

After you submit the query task, the IDs of the dependent scripts are added to the `QueryTask.inboundDependencies` property:

```

1  {"0":
2    {"type":"task.ScheduledScriptTask", "id":"SCHEDSCRIPT_0168697b126d1705061d0d690a787755500b046a1912686b10_349d94266564827c739a2ba0a5b9d476f4097217", "scriptId":"customscript_as_ftr_ss", "deploymentId":"customdeploy_ss_dpl",
3     "params":
4       {"custscript_ss_as_src_res":"SuiteScripts/ExportFile.csv"}
5   },
6  {"1":
7    {"type":"task.MapReduceScriptTask", "id":"MAPREDUCETASK_0268697b126d1705061d0d69027f7b39560f01001c_7a02acb4bdebf0103120b09302170720aa57bca4", "scriptId":"customscript_as_ftr_mr", "deploymentId":"customdeploy_mr_dpl",
8     "params":
9       {"custscript_mr_as_src_res":"SuiteScripts/ExportFile.csv"}
10  }
11 }
```

Type	Object[] (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.QueryTask
Sibling Object Members	QueryTask Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You try to set the value of this property after the <code>task.QueryTask</code> object is created.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```
1 // Add additional code
```

```

2 ...
3 // Make sure to use a script ID that represents a scheduled script in your account
4 var myScheduledScript = task.create({
5   taskType: task.TaskType.SCHEDULED_SCRIPT,
6  scriptId: 'customscript_as_ftr_ss'
7 });
8
9 // myQueryTask is an existing task.QueryTask object
10 myQueryTask.addInboundDependency(myScheduledScript);
11 ...
12 // Add additional code

```

QueryTask.query



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Query definition for the query task. The query definition is a query.Query object. To use this object, you must load the N/query module in your script and create a query using query.create(options) or load a query using query.load(options) . For more information, see N/query Module .
Type	query.Query
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.QueryTask
Sibling Object Members	QueryTask Object Members
Since	2020.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var myQueryTask = task.create({
4   taskType: task.TaskType.QUERY
5 });
6
7 // myQuery is an existing query.Query object created using the N/query module
8 myQueryTask.query = myQuery;
9 ...
10 // Add additional code

```

task.QueryTaskStatus



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Status of a query task (task.QueryTask) in the NetSuite task queue.
---------------------------	---

	You create a task.QueryTaskStatus object using task.checkStatus(options) . To check the status of a query task, you must provide the task ID of the query task. To submit a query task and obtain the task ID, use QueryTask.submit() .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Methods and Properties	QueryTaskStatus Object Members
Since	2020.2

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // myQueryTask is an existing task.QueryTask object
4 var myTaskId = myQueryTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7     taskId: myTaskId
8 });
9 ...
10 // Add additional code

```

QueryTaskStatus.fileId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID of the CSV file that query results are exported to. This property references the same CSV file that is specified by the QueryTask.fileId property or QueryTask.filePath property in the corresponding task.QueryTask object.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.QueryTaskStatus
Sibling Object Members	QueryTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY	You try to set the value of this property after the task.QueryTaskStatus object is created.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 // myQueryTask is an existing task.QueryTask object
4 var myTaskId = myQueryTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7     taskId: myTaskId
8 });
9
10 var the fileId = myTaskStatus.fileId;
11 ...
12 // Add additional code

```

QueryTaskStatus.query



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Query definition for the submitted query task. The query definition is a query.Query object. To use this object, you must load the N/query module in your script and create a query using query.create(options) or load a query using query.load(options) . For more information, see N/query Module . This property references the same query that is specified by the QueryTask.query property in the corresponding task.QueryTask object.
Type	query.Query (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.QueryTaskStatus
Sibling Object Members	QueryTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY	You try to set the value of this property after the task.QueryTaskStatus object is created.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```
1 // Add additional code
```

```

2 ...
3 //myQueryTask is an existing task.QueryTask object
4 var myTaskId = myQueryTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7     taskId: myTaskId
8 });
9
10 var theQuery = myTaskStatus.query;
11 ...
12 //Add additional code

```

QueryTaskStatus.status



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status of the submitted query task. A query task can have the following statuses, which are represented by values in the task.TaskStatus enum: <ul style="list-style-type: none"> ■ task.TaskStatus.COMPLETE ■ task.TaskStatus.FAILED ■ task.TaskStatus.PENDING ■ task.TaskStatus.PROCESSING
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.QueryTaskStatus
Sibling Object Members	QueryTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY	You try to set the value of this property after the task.QueryTaskStatus object is created.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 //myQueryTask is an existing task.QueryTask object

```

```

4  var myTaskId = myQueryTask.submit();
5
6  var myTaskStatus = task.checkStatus({
7      taskId: myTaskId
8  });
9
10 var theStatus = myTaskStatus.status;
11 ...
12 // Add additional code

```

QueryTaskStatus.taskId

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	ID of the submitted query task. This property references the same task ID that is returned by QueryTask.submit() when you submit the task for asynchronous processing.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.QueryTaskStatus
Sibling Object Members	QueryTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY	You try to set the value of this property after the task.QueryTaskStatus object is created.

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // myQueryTask is an existing task.QueryTask object
4 var myTaskId = myQueryTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7     taskId: myTaskId
8 });
9
10 var theTaskId = myTaskStatus.taskId;
11 ...
12 // Add additional code

```

task.RecordActionTask



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The properties of a record action task. Use the methods and properties for this object to submit a record action task into the task queue and to execute it asynchronously.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Methods and Properties	RecordActionTask Object Members
Since	2019.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 {
4     var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
5     recordActionTask.recordType = 'timebill';
6     recordActionTask.action = 'approve';
7     recordActionTask.params = [{recordId: 1, note: "this is a note for 1"},
8                               {recordId: 5, note: "this is a note for 5"},
9                               {recordId: 23, note: "this is a note for 23"}];
10
11    var handle = recordActionTask.submit();
12
13    var res = task.checkStatus({taskId: handle}); // returns a RecordActionTaskStatus object
14    log.debug('Initial status: ' + res.status);
15 });
16 ...
17 //Add additional code

```

RecordActionTask.submit()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Submits a record action task script deployment for processing and returns its task ID. The record action task is processed by a background process which executes the specified record action for each record ID provided in the parameters. The overall task status as well as individual action results can be queried using the <code>task.checkStatus()</code> method.
Returns	string The task ID.
Supported Script Types	Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	50 units
Module	N/task Module
Sibling Object Members	RecordActionTask Object Members
Since	2019.1

Errors

Error Code	Message	Thrown If
FAILED_TO_SUBMIT_JOB_REQUEST_1	Failed to submit job request: {reason}	The task cannot be submitted.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 {
4     var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
5     recordActionTask.recordType = 'timebill';
6     recordActionTask.action = 'approve';
7     recordActionTask.params = [{recordId: 1, note: "this is a note for 1"},
8                                {recordId: 5, note: "this is a note for 5"},
9                                {recordId: 23, note: "this is a note for 23"}];
10
11    var handle = recordActionTask.submit();
12
13    var res = task.checkStatus({taskId: handle}); // returns a RecordActionTaskStatus object
14    log.debug('Initial status: ' + res.status);
15 });
16 ...
17 //Add additional code

```

RecordActionTask.action

Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The ID of the action to be invoked.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTask Object Members
Since	2019.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 {
4     var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
5     recordActionTask.recordType = 'timebill';
6     recordActionTask.action = 'approve';
7     recordActionTask.params = [{recordId: 1, note: "this is a note for 1"},
8                                {recordId: 5, note: "this is a note for 5"},
9                                {recordId: 23, note: "this is a note for 23"}];
10
11    var handle = recordActionTask.submit();
12
13    var res = task.checkStatus({taskId: handle}); //returns a RecordActionTaskStatus object
14    log.debug('Initial status: ' + res.status);
15 };
16 ...
17 //Add additional code

```

RecordActionTask.condition



Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The condition used to select record IDs of records for which the action is to be executed. This parameter is specified with the task.ActionCondition enum. This is used in conjunction with RecordActionTask.paramCallback. If RecordActionTask.paramCallback is not specified, this default callback is used: function(v) { return { recordId: v }; }.
Type	Object
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTask Object Members
Since	2019.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.condition = task.ActionCondition.ALL_QUALIFIED_INSTANCES;
7 recordActionTask.paramCallback = function(v) {
8     return { recordId: v, note: "this is a note for " + v };
9 };

```

```

10 |     var handle = recordActionTask.submit();
11 | ...
12 | // Add additional code

```

RecordActionTask.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the task.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.RecordActionTaskStatus
Sibling Object Members	RecordActionTask Object Members
Since	

RecordActionTask.paramCallback()



Note: The content in this help topic pertains to SuiteScript 2.0.

Description	Property of type function that takes record ID and returns the parameter object for the specified record ID. Is to be used in conjunction with task.ActionCondition . This parameter cannot be specified when RecordActionTask.params is specified.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/task Module
Sibling Object Members	RecordActionTask Object Members
Since	2019.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.condition = task.ActionCondition.ALL_QUALIFIED_INSTANCES;
7 recordActionTask.paramCallback = function(v) {
8     return { recordId: v, note: "this is a note for " + v };
9 };
10 var handle = recordActionTask.submit();

```

```

11 | ...
12 | //Add additional code

```

RecordActionTask.params

Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	An array of parameter objects. Each object corresponds to one record ID of the record for which the action is to be executed. The object has the following form: {recordId: 1, someParam: 'example1', otherParam: 'example2'}
Type	Object[]
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTask Object Members
Since	2019.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 {
4     var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
5     recordActionTask.recordType = 'timebill';
6     recordActionTask.action = 'approve';
7     recordActionTask.params = [{recordId: 1, note: "this is a note for 1"},
8                                {recordId: 5, note: "this is a note for 5"},
9                                {recordId: 23, note: "this is a note for 23"}];
10
11    var handle = recordActionTask.submit();
12
13    var res = task.checkStatus({taskId: handle}); //returns a RecordActionTaskStatus object
14    log.debug('Initial status: ' + res.status);
15 });
16 ...
17 //Add additional code

```

RecordActionTask.recordType

Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The record type on which the action is to be performed. For a list of record types, see record.Type .
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/task Module
Sibling Object Members	RecordActionTask Object Members
Since	2019.1

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 {
4     var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
5     recordActionTask.recordType = 'timebill';
6     recordActionTask.action = 'approve';
7     recordActionTask.params = [{recordId: 1, note: "this is a note for 1"},
8                                {recordId: 5, note: "this is a note for 5"},
9                                {recordId: 23, note: "this is a note for 23"}];
10
11    var handle = recordActionTask.submit();
12
13    var res = task.checkStatus({taskId: handle}); //returns a RecordActionTaskStatus object
14    log.debug('Initial status: ' + res.status);
15 });
16 ...
17 //Add additional code

```

task.RecordActionTaskStatus

ℹ Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the status of a record action task.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Methods and Properties	RecordActionTaskStatus Object Members
Since	2019.1

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3     var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4     recordActionTask.recordType = 'timebill';
5     recordActionTask.action = 'approve';
6     recordActionTask.params = [{recordId: 1}, {recordId: 5}, {recordId: 23}];
7     var handle = recordActionTask.submit();
8
9 // Add any additional processing here
10
11 var taskStatus = task.checkStatus({taskId: handle}); //returns a RecordActionTaskStatus object

```

```

12 log.debug('Initial status: ' + taskStatus.status);
13 ...
14 //Add additional code
15
16 /* Example contents of a RecordActionTaskStatus object at different stages of bulk action task execution:
17
18 // initial status just after submitting the task
19 {
20     status: 'PENDING',
21     results: {},
22     errors: {},
23     complete: 0,
24     succeeded: 0,
25     failed: 0,
26     pending: 3
27 }
28
29 //in the middle of processing, two records processed, one to go
30 {
31     status: 'PROCESSING',
32     results: {
33         1: { response: { approvedId: 1 }, notifications: [] },
34         5: { response: { approvedId: 5 }, notifications: [{ title: 'Title', message: 'Message', severity: { value: 2, label: 'Warning' } }] }
35     },
36     errors: {},
37     complete: 2,
38     succeeded: 2,
39     failed: 0,
40     pending: 1
41 }
42
43 //complete, all successful
44 {
45     status: 'COMPLETE',
46     results: {
47         1: { response: { approvedId: 1 }, notifications: [] },
48         5: { response: { approvedId: 5 }, notifications: [{ title: 'Title', message: 'Message', severity: { value: 2, label: 'Warning' } }] },
49         23: { response: { approvedId: 23 }, notifications: [] }
50     },
51     errors: {},
52     complete: 3,
53     succeeded: 3,
54     failed: 0,
55     pending: 0
56 }
57
58 //complete, one action returned an error
59 {
60     status: 'COMPLETE',
61     results: {
62         1: { response: { approvedId: 1 }, notifications: [] },
63         5: { response: { approvedId: 5 }, notifications: [{ title: 'Title', message: 'Message', severity: { value: 2, label: 'Warning' } }] }
64     },
65     errors: {
66         23: { name: 'SSS_RECORD_DOES_NOT_SATISFY_CONDITION', message: ... }
67     },
68     complete: 3,
69     succeeded: 2,
70     failed: 1,
71     pending: 0
72 }
73 */
74
75 ...
76 //Add additional code

```

RecordActionTaskStatus.complete

Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The number of record actions that are already executed, either failed or successful.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTaskStatus Object Members
Since	2019.1

Errors

Error Code	Thrown If
READ_ONLY	Setting the property is attempted.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.params = [{recorderId: 1}, {recorderId: 2}];
7 var handle = recordActionTask.submit();
8
9 var taskStatus = task.checkStatus({taskId: handle}); //returns a RecordActionTaskStatus object
10 log.debug('Actions already complete: ' + taskStatus.complete);
11 // will log e.g. the following:
12 // Actions already complete: 2
13 ...
14 // Add additional code

```

RecordActionTaskStatus.errors

Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The error details of failed action executions. The value of the property is the record instance ID and the corresponding error details. The error details are returned in an unnamed object with two properties: code and message.
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/task Module
Sibling Object Members	RecordActionTaskStatus Object Members
Since	2019.1

Errors

Error Code	Thrown If
READ_ONLY	Setting the property is attempted.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.params = [{recordId: 1}, {recordId: 2}];
7 var handle = recordActionTask.submit();
8
9 var taskStatus = task.checkStatus({taskId: handle}); // returns a RecordActionTaskStatus object
10 log.debug(taskStatus.errors);
11 // will log e.g. the following:
12 // { 2: { name: 'SSS_RECORD_DOES_NOT_SATISFY_CONDITION', message: '...' } }
13 ...
14 // Add additional code

```

RecordActionTaskStatus.failed

Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The number of record actions with a failed status.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTaskStatus Object Members
Since	2019.1

Errors

Error Code	Thrown If
READ_ONLY	Setting the property is attempted.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.params = [{recorderId: 1}, {recorderId: 2}];
7 var handle = recordActionTask.submit();
8
9 var taskStatus = task.checkStatus({taskId: handle}); // returns a RecordActionTaskStatus object
10 log.debug('Actions failed: ' + taskStatus.failed);
11 // will log e.g. the following:
12 // Actions failed: 0
13 ...
14 // Add additional code

```

RecordActionTaskStatus.pending



Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The number of record actions with a pending status.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTaskStatus Object Members
Since	2019.1

Errors

Error Code	Thrown If
READ_ONLY	Setting the property is attempted.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.params = [{recorderId: 1}, {recorderId: 2}];
7 var handle = recordActionTask.submit();
8
9 var taskStatus = task.checkStatus({taskId: handle}); // returns a RecordActionTaskStatus object
10 log.debug('Actions pending: ' + taskStatus.pending);
11 // will log e.g. the following:

```

```

12 // Actions pending: 2
13 ...
14 // Add additional code

```

RecordActionTaskStatus.results



Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The results of successfully executed record action tasks. The value of the property is the task instance ID and the corresponding action result.
Type	Object (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTaskStatus Object Members
Since	2019.1

Errors

Error Code	Thrown If
READ_ONLY	Setting the property is attempted.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.params = [{recordId: 1}, {recordId: 2}];
7 var handle = recordActionTask.submit();
8
9 var taskStatus = task.checkStatus({taskId: handle}); // returns a RecordActionTaskStatus object
10 log.debug(taskStatus.results);
11 // will log e.g. the following:
12 // { 1: { response: { approved: true }, notifications: [] } }
13 ...
14 // Add additional code

```

RecordActionTaskStatus.status



Note: The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	Represents the record action task status. Returns a value from the task.TaskStatus enum.
Type	string (read-only)

Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTaskStatus Object Members
Since	2019.1

Errors

Error Code	Thrown If
READ_ONLY	Setting the property is attempted.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.params = [{recorderId: 1}, {recorderId: 2}];
7 var handle = recordActionTask.submit();
8
9 var taskStatus = task.checkStatus({taskId: handle}); //returns a RecordActionTaskStatus object
10 log.debug('Current task status: ' + taskStatus.status);
11 // will log e.g. the following:
12 // Current task status: PENDING
13 ...
14 // Add additional code

```

RecordActionTaskStatus.succeeded

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Parameter Description	The number of record actions with a successful status.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Object Members	RecordActionTaskStatus Object Members
Since	2019.1

Errors

Error Code	Thrown If
READ_ONLY	Setting the property is attempted.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.params = [{recordId: 1}, {recordId: 2}];
7 var handle = recordActionTask.submit();
8
9 var taskStatus = task.checkStatus({taskId: handle}); // returns a RecordActionTaskStatus object
10 log.debug('Actions executed successfully: ' + taskStatus.succeeded);
11 // will log e.g. the following:
12 // Actions executed successfully: 1
13 ...
14 // Add additional code

```

RecordActionTaskStatus.taskId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The task ID associated with the specified task.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.RecordActionTaskStatus
Sibling Object Members	RecordActionTaskStatus Object Members
Since	

Errors

Error Code	Error Message	Thrown If
READ_ONLY		Setting the property is attempted.

task.ScheduledScriptTask



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	All the properties of scheduled script task in SuiteScript. Use this object to place a scheduled script deployment into the NetSuite scheduling queue. To use the ScheduledScriptTask Object: <ol style="list-style-type: none"> 1. In the NetSuite UI, create the script record and script deployment record.
---------------------------	--

	<ol style="list-style-type: none"> 2. Use task.create(options) to create the ScheduledScriptTask object. 3. Use the ScheduledScriptTask object properties to set the script and deployment properties. 4. Use ScheduledScriptTask.submit() to deploy the scheduled script to the NetSuite scheduling queue. 5. Use the properties for the task.ScheduledScriptTaskStatus object to get the status of the scheduled script. <p>For a complete list of this object's methods and properties, see ScheduledScriptTask Object Members.</p> <p>For more information about scheduled scripts in NetSuite, see the help topic SuiteScript 2.0 Scheduled Script Type.</p>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var scriptTask = task.create({taskType: task.TaskType.SCHEDULED_SCRIPT});
4 scriptTask.scriptId = 1234;
5 scriptTask.deploymentId = 'customdeploy1';
6 scriptTask.params = {searchId: 'custsearch_456'};
7 var scriptTaskId = scriptTask.submit();
8 ...
9 //Add additional code

```

ScheduledScriptTask.submit()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Directs NetSuite to place a scheduled script deployment into the NetSuite scheduling queue and returns a unique ID for the task.</p> <p>Additionally, note the following:</p> <ul style="list-style-type: none"> ■ The scheduled script must have a status of Not Scheduled on the Script Deployment page. If the script status is set to Testing on the Script Deployment page, this method will not place the script into the scheduling queue. ■ If the deployment status on the Script Deployment page is set to Scheduled, the script will be placed into the queue according to the time(s) specified on the Script Deployment page. ■ Only roles with the SuiteScript Scheduling permission or Administrators can run scheduled scripts executed by API. If a user event script calls ScheduledScriptTask.submit(), the user event script must be initiated by a role with permission. ■ A scheduled script can be submitted for processing only if there is no unfinished scheduled script task for the same script ID and script deployment ID. Therefore, if a scheduled script resubmits itself, the actual resubmit does not occur until the current execution completes. This delay is necessary to avoid the existence of two unfinished tasks for the
---------------------------	--

	same deployment of the same script. For this reason, if a scheduled script uses the submit() method to resubmit itself, then at runtime, no task ID is returned when the scheduled script is submitted.
Returns	string The task ID as a string, except as noted above.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
FAILED_TO_SUBMIT_JOB_REQUEST_1	Failed to submit job request: {reason}	Task cannot be submitted.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var scheduledScriptTaskId = scriptTask.submit();
4 ...
5 //Add additional code

```

ScheduledScriptTask.deploymentId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Script ID (as a string), for the script deployment record associated with a task.ScheduledScriptTask Object.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```
1 //Add additional code.
```

```

2 ..
3 scheduledTask.deploymentId = custdeploy1;
4 ...
5 //Add additional code

```

ScheduledScriptTask.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the task.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.ScheduledScriptTask
Sibling Object Members	ScheduledScriptTask Object Members
Since	2015.2

ScheduledScriptTask.params



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Object with key/value pairs that override static script parameter field values on the script deployment. Use these parameters for the task.ScheduledScriptTask object to programmatically pass values to the script deployment. For more information about script parameters, see the help topic Creating Script Parameters Overview .
Type	Object
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 scriptTask.params = {searchId: 'custsearch_456'};
4 ...
5 //Add additional code

```

ScheduledScriptTask.scriptId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID (as a number), or script ID (as a string), for the script record associated with a task.ScheduledScriptTask object.
Type	number string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var scheduledscriptId = 34;
4 ...
5 //Add additional code

```

task.ScheduledScriptTaskStatus



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The properties and status of a scheduled script placed into the NetSuite scheduling queue. Use task.checkStatus(options) with the unique ID for the scheduled script task to get the ScheduledScriptTaskStatus Object. For a complete list of this object's properties, see ScheduledScriptTaskStatus Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var res = task.checkStatus(scriptTaskId);

```

```

4 log.debug('Initial status: ' + res.status);
5 ...
6 //Add additional code

```

ScheduledScriptTaskStatus.deploymentId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Script ID for a script deployment record associated with a specific task.ScheduledScriptTask Object. Use this ID to get more details about the script deployment record for the scheduled task.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 log.audit({
4   details:'Deployment ID: ' + status.scriptId
5 });
6 ...
7 //Add additional code

```

ScheduledScriptTaskStatus.scriptId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID for a script record associated with a specific task.ScheduledScriptTask Object. Use this ID to get more details about the script record for the scheduled task.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .

Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 log.audit('Initial status: ' + status.scriptId);
4 ...
5 //Add additional code

```

ScheduledScriptTaskStatus.status

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status for a scheduled script task. Returns a task.TaskStatus enum value.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 log.audit({
4   details: 'Status: ' + summary.status
5 });
6 ...
7 //Add additional code

```

ScheduledScriptTaskStatus.taskId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The task ID associated with the specified task.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.ScheduledScriptTaskStatus
Sibling Object Members	ScheduledScriptTaskStatus Object Members
Since	

Errors

Error Code	Error Message	Thrown If
READ_ONLY		Setting the property is attempted.

task.SearchTask

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>The properties of a search task. Use the methods and properties for this object to submit a search task into the task queue, execute it asynchronously, and persist search results. Similar to SuiteAnalytics persisted search functionality, this capability is useful for searches across high volumes of data.</p> <p>You can create a <code>task.SearchTask</code> object using task.create(options).</p> <p>Use the <code>task.SearchTask</code> object to do the following:</p> <ul style="list-style-type: none"> ■ Set the search ID using the SearchTask.savedSearchId property. ■ Set the file ID or file path of a CSV file in the File Cabinet. Search results are exported to this file. Use the SearchTask.fileId property or the SearchTask.filePath property. Exactly one of these properties must be set. If both are set, an error occurs. ■ Add dependent scripts to the search task using SearchTask.addInboundDependency(). Dependent scripts are processed automatically when the search task is complete. ■ Submit the search task to the NetSuite task queue using SearchTask.submit(). ■ Get the status of a search task using the properties of the task.SearchTaskStatus object. <p>Note: There is a limit to the number of asynchronous searches running at the same time. The limit is set to be the same as the limit for CSV import. The file size limit is based on File Cabinet limits.</p>
Supported Script Types	<p>Server scripts For more information, see the help topic SuiteScript 2.0 Script Types.</p>

Module	N/task Module
Since	2017.1

Syntax

```

1 //Add additional code
2 ...
3 var searchTask = task.create({
4     taskType: task.TaskType.SEARCH
5 });
6 searchTask.savedSearchId = 51;
7
8 var path = 'ExportFolder/export.csv';
9 searchTask.filePath = path;
10
11 var searchTaskId = searchTask.submit();
12 ...
13 //Add additional code

```

SearchTask.addInboundDependency()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a scheduled script task (task.ScheduledScriptTask) or map/reduce script task (task.MapReduceScriptTask) to the search task as a dependent script. Dependent scripts are processed automatically when the search task is complete. For more information, see the help topic SuiteCloud Processors .
	 Note: You can add only scheduled scripts or map/reduce scripts as dependent scripts to asynchronous search tasks. Other script types are not supported.
	When you use this method to add a dependent script, the script is considered an inbound dependency of the search task. The added script depends on the search task. For example, if you add a scheduled script task to a search task as a dependent script, the scheduled script depends on the search task. Because <code>addInboundDependency()</code> is called on the search task, any dependent scripts that you add are considered inbound dependencies.
Returns	void
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/task Module
Since	2018.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
dependentScript	task.ScheduledScriptTask task.MapReduceScriptTask	Required	The script to add as a dependent script to the search task.	2018.2

Parameter	Type	Required / Optional	Description	Since
			<p>Use <code>task.create(options)</code> and the <code>task.TaskType</code> enum to create a script task with a type of <code>SCHEDULED_SCRIPT</code> or <code>MAP_REDUCE</code>. This script task is a <code>task.ScheduledScriptTask</code> object or a <code>task.MapReduceScriptTask</code>, and you can add this script task as a dependent script to the search task. The dependent script is processed when the search task is complete.</p> <p>You can add only one dependent script per call to <code>SearchTask.addInboundDependency()</code>.</p>	

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var scheduledScript = task.create({
4   taskType: task.TaskType.SCHEDULED_SCRIPT
5 });
6 // Set the properties of the scheduled script task
7 scheduledScript.scriptId = 'customscript_as_ftr_ss';
8 ...
9
10 var asyncTask = task.create({
11   taskType: task.TaskType.SEARCH
12 });
13 // Set the properties of the search task
14 asyncTask.savedSearchId = 'customsearch35';
15 ...
16
17 asyncTask.addInboundDependency(scheduledScript);
18
19 var taskId = asyncTask.submit();
20 ...
21 // Add additional code

```

SearchTask.submit()

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Directs NetSuite to initiate the asynchronous search task and return a unique ID for the task. When the submission is successful, this method adds the internal IDs of any dependent scripts (added using <code>SearchTask.addInboundDependency()</code>) to the <code>SearchTask.inboundDependencies</code> property. Use <code>task.SearchTaskStatus</code> to view the status of a submitted task.
Returns	string The task ID
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	100 units

Module	N/task Module
Since	2017.1

Errors

Error Code	Thrown If
FAILED_TO_SUBMIT_JOB_REQUEST_1	Task cannot be submitted.
SSS_MISSING_REQD_ARGUMENT	A required parameter is missing.
YOU_DO_NOT_HAVE_ACCESS_TO_THE_MEDIA_ITEM_YOU_SELECTED	You do not have permission to access the file.
THAT_RECORD_DOES_NOT_EXIST	The file Object references a file that doesn't exist.
MUST_IDENTIFY_A_FILE	The path specifies a folder and not a file.
CANNOT_RESUBMIT_SUBMITTED_ASYNC_SEARCH_TASK	The search task was already submitted and completed successfully.
ASYNC_SEARCH_DEPENDENCY_MR_ALREADY_SUBMITTED	A dependent map/reduce script is already submitted and is not complete.
ASYNC_SEARCH_DEPENDENCY_MR_INCORRECT_STATUS	The status of the deployment record for the specified dependent map/reduce script has a value other than "Not Scheduled".
ASYNC_SEARCH_DEPENDENCY_SS_ALREADY_SUBMITTED	A dependent scheduled script is already submitted and is not complete.
ASYNC_SEARCH_DEPENDENCY_SS_INCORRECT_STATUS	The status of the deployment record for the specified dependent scheduled script has a value other than "Not Scheduled".
ASYNC_SEARCH_DEPLOYMENT_FOR_DEPENDENCY	A deployment record for the specified dependent script is not available for one of the following reasons: <ul style="list-style-type: none"> ■ A deployment record was not specified when the dependent script was created, and automatic lookup for an available deployment record failed. ■ The deployment record specified when the dependent script was created is not found.
ASYNC_SEARCH_MULTIPLE_DEPENDENCIES	The same dependent script is passed to this method more than once.
ASYNC_SEARCH_SCRIPT_ID_NOT_FOUND	The specified dependent script is not found.
ASYNC_SEARCH_SEARCH_ID_NOT_FOUND	The search task with the specified search ID is not found.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

1 | //Add additional code

```

2 ...
3 var searchTriggerTask = searchTask.submit();
4 ...
5 //Add additional code

```

SearchTask fileId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	ID of the CSV file to export search results into.
	 Note: Either this property or the SearchTask.filePath property must be set. If both are set, an error occurs.
Type	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2017.1

Errors

Error Code	Thrown If
UNSUPPORTED_COMBINATION_OF_PARAMETERS	Both this property and the SearchTask.filePath property are set at the same time.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 searchTask.fileId = 18;
4 ...
5 //Add additional code

```

SearchTask filePath

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Path of the CSV file to export search results into.
	 Note: Either this property or the SearchTask.fileId property must be set. If both are set, an error occurs.
Type	string

Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2017.1

Errors

Error Code	Thrown If
UNSUPPORTED_COMBINATION_OF_PARAMETERS	Both this property and the SearchTask.fileId property are set at the same time.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 searchTask.filePath= 'ExportFolder/export.csv'
4 ...
5 //Add additional code

```

SearchTask.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the task.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SearchTask
Sibling Object Members	SearchTask Object Members
Since	

SearchTask.inboundDependencies



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Key/value pairs that contain information about the dependent scripts added to the search task. Use this property to verify the properties of dependent scripts after you add the scripts using SearchTask.addInboundDependency() .
-----------------------------	--

This property uses nested objects to store information about each dependent script. A nested object is included for each dependent script added to the search task. The nested object contains information such as the task type, script ID, and deployment ID. It also includes the index of the script (starting at 0). Dependent scripts are indexed in the order they are added to the search task.

For example, consider a situation in which you add a scheduled script task and a map/reduce script task to a search task as dependent scripts. After you add the dependent scripts, but before you submit the search task using [SearchTask.submit\(\)](#), the value of the `SearchTask.inboundDependencies` property is similar to the following:

```

1  {"0":
2    {"type":"task.ScheduledScriptTask", "scriptId":"customscript_as_ftr_ss", "deploymentId":"customdeploy_ss_dpl",
3     "params":
4       {"custscript_ss_as_srch_res":"SuiteScripts/ExportFile.csv"}
5   },
6  {"1":
7    {"type":"task.MapReduceScriptTask", "scriptId":"customscript_as_ftr_mr", "deploymentId":"customdeploy_mr_dpl",
8     "params":
9       {"custscript_mr_as_srch_res":"SuiteScripts/ExportFile.csv"}
10  }
11 }

```

After you submit the search task, the internal IDs of the dependent scripts are added to the `SearchTask.inboundDependencies` property:

```

1  {"0":
2    {"type":"task.ScheduledScriptTask", "id":"SCHEDSCRIPT_0168697b126d1705061d0d690a787755500b046a1912686b10_349d94266564827c739a2ba0a5b9d476f4097217", "scriptId":"customscript_as_ftr_ss", "deploymentId":"customdeploy_ss_dpl",
3     "params":
4       {"custscript_ss_as_srch_res":"SuiteScripts/ExportFile.csv"}
5   },
6  {"1":
7    {"type":"task.MapReduceScriptTask", "id":"MAPREDUSETASK_0268697b126d1705061d0d69027f7b39560f01001c_7a02acb4bdebf0103120b09302170720aa57bca4", "scriptId":"customscript_as_ftr_mr", "deploymentId":"customdeploy_mr_dpl",
8     "params":
9       {"custscript_mr_as_srch_res":"SuiteScripts/ExportFile.csv"}
10  }
11 }

```

Type	Object[] (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2018.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	Setting the property is attempted

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var scheduledScript = task.create({

```

```

4     taskType: task.TaskType.SCHEDULED_SCRIPT
5 });
6 // Set the properties of the scheduled script task
7 scheduledScript.scriptId = 'customscript_as_ftr_ss';
8 ...
9
10 var mapReduceScript = task.create({
11     taskType: task.TaskType.MAP_REDUCE
12 });
13 // Set the properties of the map/reduce script task
14 mapReduceScript.scriptId = 'customscript_as_ftr_mr';
15 ...
16
17 asyncTask.addInboundDependency(scheduledScript);
18 asyncTask.addInboundDependency(mapReduceScript);
19
20 var asyncTaskId = asyncTask.submit();
21
22 // Iterate over the dependent scripts
23 var p = asyncTask.inboundDependencies;
24 for (var key in p) {
25     log.debug(key + ' > ' + p[key]);
26 }
27 ...
28 // Add additional code

```

SearchTask.savedSearchId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	ID of the saved search to be executed during the task.
Type	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2017.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 searchTask.savedSearchId = 51;
4 ...
5 //Add additional code

```

task.SearchTaskStatus

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Object Description	The status of an asynchronous search task (task.SearchTask) placed into the NetSuite task queue. To initiate the task and retrieve the task ID, use SearchTask.submit() .
---------------------------	---

Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2017.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var searchTaskStatus = task.checkStatus({
4     taskId: 51
5 });
6
7 if (searchTaskStatus.status === task.TaskStatus.FAILED) {
8     // Handle the task failure
9 }
10 ...
11 // Add additional code

```

SearchTaskStatus.fileId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	ID of CSV file into which search results are exported.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2017.1

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var status = task.checkStatus({
4     searchTaskId: 81
5 });

```

```

6 log.audit({
7   title: 'File ID',
8   details: status.fileId
9 });
10 ...
11 //Add additional code

```

SearchTaskStatus.savedSearchId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the saved search executed during the task.
Type	number (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2017.1

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var status = task.checkStatus({
4   searchTaskId: 81
5 });
6 log.audit({
7   title: 'Saved Search ID',
8   details: status.savedSearchId
9 });
10 ...
11 //Add additional code

```

SearchTaskStatus.status

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status for an asynchronous search placed in the NetSuite task queue by SearchTask.submit() . Returns a task.TaskStatus enum value.
Type	string (read-only)
Supported Script Types	Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2017.1

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 log.audit({
5   title: 'Status',
6   details: summary.status
7 });
8 ...
9 ...
10 //Add additional code

```

SearchTaskStatus.taskId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	ID of the task.SearchTask Object. Use SearchTask.submit() to return this ID.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2017.1

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```
1 //Add additional code
```

```

2 ...
3 var searchTaskId = searchTask.submit();
4 ...
5 //Add additional code

```

task.SuiteQLTask



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>An asynchronous SuiteQL task. Use the methods and properties for this object to submit a SuiteQL task to the NetSuite task queue and execute it asynchronously.</p> <p>You can create a task.SuiteQLTask object using task.create(options). You can populate the properties of this object as you create it. With a task.SuiteQLTask object, you can do the following:</p> <ul style="list-style-type: none"> ▪ Specify the SuiteQL query to submit using the SuiteQLTask.query property. This property accepts a string that represents a SuiteQL query. For more information, see the help topic SuiteQL. ▪ Provide parameters to use in the SuiteQL query using the SuiteQLTask.params property. ▪ Set the file ID or file path of a CSV file in the File Cabinet. SuiteQL query results are exported to this file. Use the SuiteQLTask fileId property to specify a file ID, or use the SuiteQLTask.filePath property to specify a file path. Only one of these properties can be set at the same time. If both are set, an error occurs. ▪ Add dependent tasks to the SuiteQL task using SuiteQLTask.addInboundDependency(options). Dependent tasks are processed automatically when the SuiteQL task is complete. ▪ Submit the SuiteQL task to the task queue using SuiteQLTask.submit(). ▪ Get the status of a SuiteQL task using the properties of an associated task.SuiteQLTaskStatus object. The task.checkStatus(options) method returns a task.SuiteQLTaskStatus object when you specify a task ID that represents a SuiteQL task.
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Methods and Properties	SuiteQLTask Object Members
Since	2020.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var mySuiteQLTask = task.create({
4   taskType: task.TaskType.SUITE_QL
5 });
6 ...
7 // Add additional code

```

SuiteQLTask.addInboundDependency(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Adds a scheduled script task or map/reduce script task as a dependent task to the SuiteQL task. Dependent tasks are processed automatically using SuiteCloud Processors when the SuiteQL task is complete. For more information, see the help topic SuiteCloud Processors.</p> <p>You can add a dependent task in the following ways:</p> <ul style="list-style-type: none"> ■ Provide a <code>task.ScheduledScriptTask</code> object or <code>task.MapReduceScriptTask</code> object that represents the dependent task as the only parameter to this method. ■ Provide an Object as the only parameter that includes values for the following properties: <ul style="list-style-type: none"> □ <code>taskType</code> □ <code>scriptId</code> □ <code>deploymentId</code> (optional) □ <code>params</code> (optional) <p>For descriptions of these properties, see Parameters.</p> <div data-bbox="447 823 1385 960" style="background-color: #ffffcc; border: 1px solid #ffcc00; padding: 10px;"> <p>Important: You can add only scheduled scripts or map/reduce scripts as dependent tasks to asynchronous SuiteQL tasks. Other script types are not supported. You can add only one dependent task per call to <code>SuiteQLTask.addInboundDependency(options)</code>.</p> </div> <p>When you use this method to add a dependent task, the added task is considered an inbound dependency of the SuiteQL task. For example, if you add a scheduled script task to a SuiteQL task as a dependent task, the scheduled script depends on the SuiteQL task. Because <code>SuiteQLTask.addInboundDependency(options)</code> is called on the SuiteQL task, any dependent task that you add is considered an inbound dependency.</p>
Returns	<code>void</code>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/task Module
Parent Object	<code>task.SuiteQLTask</code>
Sibling Object Members	SuiteQLTask Object Members
Since	2020.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.scriptId</code>	string	required	The script ID of the scheduled script record or map/reduce script record for the dependent task.

Parameter	Type	Required / Optional	Description
options.taskType	string	required	<p>The type of dependent task.</p> <p>This property uses one of the following values in the task.TaskType enum:</p> <ul style="list-style-type: none"> ▪ <code>task.TaskType.SCHEDULED_SCRIPT</code> ▪ <code>task.TaskType.MAP_REDUCE</code>
options.deploymentId	string	optional	<p>The script ID of the script deployment record for the dependent task.</p> <p>To use this property, the scheduled script or map/reduce script for the dependent task must be deployed. If you do not specify a value for this property, an available deployment is selected automatically.</p>
options.params	Object	optional	<p>The parameters for the scheduled script or map/reduce script.</p> <p>To use this property, you must first create script parameters that are required for the dependent task. For example, if your dependent task requires a reference to the CSV result file, you must create a script parameter that maps the CSV result file ID to the file name, as in the following example:</p> <pre> 1 { 2 'custscript_query_res' : 'result 3 File.csv' 4 }</pre> <p>For more information about script parameters, see the help topic Creating Script Parameters (Custom Fields).</p>

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // Make sure to use a script ID that represents a scheduled script in your account
4 var myScheduledScript = task.create({
5   taskType: task.TaskType.SCHEDULED_SCRIPT,
6  scriptId: 'customscript_as_ftr_ss'
7 });
8
9 // mySuiteQLTask is an existing task.SuiteQLTask object
10 mySuiteQLTask.addInboundDependency(myScheduledScript);
11 ...
12 // Add additional code
```

SuiteQLTask.submit()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Submits the SuiteQL task for asynchronous processing. This method returns a unique ID for the SuiteQL task. When the submission is successful, this method adds the script IDs of any dependent tasks (added using SuiteQLTask.addInboundDependency(options)) to the SuiteQLTask.inboundDependencies property. Use the task.SuiteQLTaskStatus object to view the status of a submitted SuiteQL task. The task.checkStatus(options) method returns a task.SuiteQLTaskStatus object when you specify a task ID that represents a SuiteQL task.
Returns	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	100 units
Module	N/task Module
Parent Object	task.SuiteQLTask
Sibling Object Members	SuiteQLTask Object Members
Since	2020.2

Errors

Error Code	Thrown If
ASYNC_SUITEQL_DEPENDENCY_MR_ALREADY_SUBMITTED	A dependent map/reduce script task is already submitted and is not complete.
ASYNC_SUITEQL_DEPENDENCY_MR_INCORRECT_STATUS	The status of the script deployment record for the specified dependent map/reduce script task has a value other than "Not Scheduled".
ASYNC_SUITEQL_DEPENDENCY_SS_ALREADY_SUBMITTED	A dependent scheduled script task is already submitted and is not complete.
ASYNC_SUITEQL_DEPENDENCY_SS_INCORRECT_STATUS	The status of the script deployment record for the specified dependent scheduled script task has a value other than "Not Scheduled".
ASYNC_SUITEQL_DEPLOYMENT_FOR_DEPENDENCY	A script deployment record for the specified dependent script task is not available for one of the following reasons: <ul style="list-style-type: none"> ■ A script deployment record was not specified when the dependent task was created, and automatic lookup for an available script deployment record failed. ■ The script deployment record specified when the dependent task was created is not found.

Error Code	Thrown If
ASYNC_SUITEQL_MULTIPLE_DEPENDENCIES	The same dependent task is passed to this method more than once.
ASYNC_SUITEQL_SCRIPT_ID_NOT_FOUND	The specified dependent task is not found.
ASYNC_SUITEQL_SUITEQL_ID_NOT_FOUND	A SuiteQL task with the specified script ID is not found.
CANNOT_RESUBMIT_SUBMITTED_ASYNC_SUITEQL_TASK	The SuiteQL task was already submitted and completed successfully.
FAILED_TO_SUBMIT_JOB_REQUEST_1	The SuiteQL task cannot be submitted due to an unexpected error.
MUST_IDENTIFY_A_FILE	The SuiteQLTask.filePath property specifies a folder and not a file.
SSS_MISSING_REQD_ARGUMENT	A required property is not specified.
THAT_RECORD_DOES_NOT_EXIST	The SuiteQLTask fileId property or SuiteQLTask.filePath property references a file that does not exist.
YOU_DO_NOT_HAVE_ACCESS_TO_THE_MEDIA_ITEM_YOU_SELECTED	You do not have permission to access the file specified by the SuiteQLTask fileId property or SuiteQLTask.filePath .

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var mySuiteQLTask = task.create({
4   taskType: task.TaskType.SUITE_QL
5 });
6
7 // Set the properties of the mySuiteQLTask object
8
9 // Submit the task
10 var myTaskId = mySuiteQLTask.submit();
11
12 // Check the task status
13 var myStatus = task.checkStatus({
14   taskId: myTaskId
15 });
16 ...
17 // Add additional code

```

SuiteQLTask.fileId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID of the CSV file to export SuiteQL query results to. The CSV file must be located in the File Cabinet. You can provide a value for this property or the SuiteQLTask.filePath property. Only one of these properties can be set at the same time. If both are set, an error occurs.
-----------------------------	--

Type	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTask
Sibling Object Members	SuiteQLTask Object Members
Since	2020.2

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You specify values for both SuiteQLTask.fileId and SuiteQLTask.filePath.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var mySuiteQLTask = task.create({
4   taskType: task.TaskType.SUITE_QL
5 });
6
7 mySuiteQLTask.filePath = "C:/temp/test.csv";
8 ...
9 // Add additional code

```

SuiteQLTask.filePath

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Path of the CSV file to export SuiteQL query results to. The CSV file must be located in the File Cabinet. You can provide a value for this property or the SuiteQLTask.fileId property. Only one of these properties can be set at the same time. If both are set, an error occurs.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTask
Sibling Object Members	SuiteQLTask Object Members

Since	2020.2
-------	--------

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You specify values for both SuiteQLTask.fileId and SuiteQLTask.filePath.

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 var mySuiteQLTask = task.create({
4   taskType: task.TaskType.SUITE_QL
5 });
6
7 mySuiteQLTask.filePath = 'ExportFolder/export.csv';
8 ...
9 // Add additional code

```

SuiteQLTask.inboundDependencies

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>Key-value pairs that contain information about the dependent tasks added to the SuiteQL task. Use this property to verify the properties of dependent tasks after you add the tasks using SuiteQLTask.addInboundDependency(options).</p> <p>This property uses nested objects to store information about each dependent task. A nested object is included for each dependent task added to the SuiteQL task, and these objects are referenced by their index (starting at 0). Dependent tasks are indexed in the order they are added to the SuiteQL task. Each nested object contains the task type, script ID, script deployment ID, and script parameters.</p> <p>For example, consider a situation in which you add a scheduled script task and a map/reduce script task to a SuiteQL task as dependent tasks. After you add the dependent tasks but before you submit the SuiteQL task using QueryTask.submit(), the value of the SuiteQLTask.inboundDependencies property is similar to the following:</p> <pre> 1 {"0": 2 {"type":"task.ScheduledScriptTask", "scriptId":"customscript_as_ftr_ss", "deploymentId":"customdeploy_ss_dpl", 3 "params": 4 {"custscript_ss_as_srch_res":"SuiteScripts/ExportFile.csv"} 5 }, 6 "1": 7 {"type":"task.MapReduceScriptTask", "scriptId":"customscript_as_ftr_mr", "deploymentId":"customdeploy_mr_dpl", 8 "params": 9 {"custscript_mr_as_srch_res":"SuiteScripts/ExportFile.csv"} 10 } 11 } </pre> <p>After you submit the SuiteQL task, the IDs of the dependent scripts are added to the SuiteQLTask.inboundDependencies property:</p> <pre> 1 {"0": </pre>
-----------------------------	--

	<pre> 2 {"type":"task.ScheduledScriptTask", "id":"SCHEDSCRIPT_0168697b126d1705061d0d690a787755500b046a1912686b10_349d94266564827c739a2ba0a5b9d476f4097217", "scriptId":"customscript_as_ftr_ss", "deploymentId":"customdeploy_ss_dpl", 3 "params": 4 {"custscript_ss_as_srch_res":"SuiteScripts/ExportFile.csv"} 5 }, 6 "1": 7 {"type":"task.MapReduceScriptTask", "id":"MAPREDUCETASK_0268697b126d1705061d0d69027f7b39560f01001c_7a02acb4bdebf0103120b09302170720aa57bca4", "scriptId":"customscript_as_ftr_mr", "deploymentId":"customdeploy_mr_dpl", 8 "params": 9 {"custscript_mr_as_srch_res":"SuiteScripts/ExportFile.csv"} 10 } 11 } </pre>
Type	Object[] (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTask
Sibling Object Members	SuiteQLTask Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You try to set the value of this property after the task.SuiteQLTask object is created.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // Make sure to use a script ID that represents a scheduled script in your account
4 var myScheduledScript = task.create({
5   taskType: task.TaskType.SCHEDULED_SCRIPT,
6   scriptId: 'customscript_as_ftr_ss'
7 });
8 ...
9 // mySuiteQLTask is an existing task.SuiteQLTask object
10 mySuiteQLTask.addInboundDependency(myScheduledScript);
11 ...
12 // Add additional code

```

SuiteQLTask.params



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Parameters for the SuiteQL query. The specified parameters are substituted into the SuiteQL query string (represented by SuiteQLTask.query) when you submit the SuiteQL task using SuiteQLTask.submit() .
----------------------	---

Type	Array<string boolean number>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTask
Sibling Object Members	SuiteQLTask Object Members
Since	2020.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var mySuiteQLTask = task.create({
4   taskType: task.TaskType.SUITE_QL
5 });
6
7 mySuiteQLTask.params = ['myStringParameter', true];
8 ...
9 // Add additional code

```

SuiteQLTask.query



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	SuiteQL query definition for the SuiteQL task. The SuiteQL query definition is a string that represents a SuiteQL query. For more information, see the help topic SuiteQL .
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTask
Sibling Object Members	SuiteQLTask Object Members
Since	2020.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```
1 // Add additional code
```

```

2 ...
3 var mySuiteQLTask = task.create({
4   taskType: task.TaskType.SUITE_QL
5 });
6
7 mySuiteQLTask.query = 'SELECT customer.entityid, customer.email FROM customer';
8 ...
9 //Add additional code

```

task.SuiteQLTaskStatus



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The status of an asynchronous SuiteQL task (task.SuiteQLTask) in the NetSuite task queue. You create a <code>task.SuiteQLTaskStatus</code> object using task.checkStatus(options) . To check the status of a SuiteQL task, you must provide the task ID of the SuiteQL task. To submit a SuiteQL task and obtain the task ID, use SuiteQLTask.submit() .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Methods and Properties	SuiteQLTaskStatus Object Members
Since	2020.2

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // mySuiteQLTask is an existing task.SuiteQLTask object
4 var myTaskId = mySuiteQLTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7   taskId: myTaskId
8 });
9 ...
10 // Add additional code

```

SuiteQLTaskStatus.fileId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID of the CSV file that SuiteQL query results are exported to. This property references the same CSV file that is specified by the SuiteQLTask.fileId property or SuiteQLTask.filePath property in the corresponding <code>task.SuiteQLTask</code> object.
Type	number (read-only)
Supported Script Types	Server scripts

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTaskStatus
Sibling Object Members	SuiteQLTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY	You try to set the value of this property after the task.SuiteQLTaskStatus object is created.

Syntax

⚠ Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // mySuiteQLTask is an existing task.SuiteQLTask object
4 var myTaskId = mySuiteQLTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7     taskId: myTaskId
8 });
9
10 var the fileId = myTaskStatus.fileId;
11 ...
12 // Add additional code

```

SuiteQLTaskStatus.params

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Parameters for the SuiteQL query. The specified parameters are substituted into the SuiteQL query string (represented by SuiteQLTask.query) when you submit the SuiteQL task using SuiteQLTask.submit() . This property references the same set of parameters that are specified by the SuiteQLTask.params property in the corresponding task.SuiteQLTask object.
Type	Array<string boolean number> (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTaskStatus
Sibling Object Members	SuiteQLTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You try to set the value of this property after the task.SuiteQLTaskStatus object is created.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // mySuiteQLTask is an existing task.SuiteQLTask object
4 var myTaskId = mySuiteQLTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7     taskId: myTaskId
8 });
9
10 var theParams = myTaskStatus.params;
11 ...
12 // Add additional code

```

SuiteQLTaskStatus.query

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	SuiteQL query definition for the SuiteQL task. The SuiteQL query definition is a string that represents a SuiteQL query. For more information, see the help topic SuiteQL . This property references the same query that is specified by the SuiteQLTask.query property in the corresponding task.SuiteQLTask object.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTaskStatus
Sibling Object Members	SuiteQLTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY_PROPERTY	You try to set the value of this property after the task.SuiteQLTaskStatus object is created.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 // mySuiteQLTask is an existing task.SuiteQLTask object
4 var myTaskId = mySuiteQLTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7     taskId: myTaskId
8 });
9
10 var theQuery = myTaskStatus.query;
11 ...
12 // Add additional code

```

SuiteQLTaskStatus.status



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status of the SuiteQL task. A SuiteQL task can have the following statuses, which are represented by values in the task.TaskStatus enum: <ul style="list-style-type: none"> ■ task.TaskStatus.COMPLETE ■ task.TaskStatus.FAILED ■ task.TaskStatus.PENDING ■ task.TaskStatus.PROCESSING
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTaskStatus
Sibling Object Members	SuiteQLTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY	You try to set the value of this property after the task.SuiteQLTaskStatus object is created.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```
1 // Add additional code
```

```

2 ...
3 // mySuiteQLTask is an existing task.SuiteQLTask object
4 var myTaskId = mySuiteQLTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7   taskId: myTaskId
8 });
9
10 var theStatus = myTaskStatus.status;
11 ...
12 // Add additional code

```

SuiteQLTaskStatus.taskId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	ID of the submitted SuiteQL task. This property references the same task ID that is returned by SuiteQLTask.submit() when you submit the task for asynchronous processing.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.SuiteQLTaskStatus
Sibling Object Members	SuiteQLTaskStatus Object Members
Since	2020.2

Errors

Error Code	Thrown If
READ_ONLY	You try to set the value of this property after the task.SuiteQLTaskStatus object is created.

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 // mySuiteQLTask is an existing task.SuiteQLTask object
4 var myTaskId = mySuiteQLTask.submit();
5
6 var myTaskStatus = task.checkStatus({
7   taskId: myTaskId
8 });
9
10 var theTaskId = myTaskStatus.taskId;
11 ...
12 // Add additional code

```

task.WorkflowTriggerTask



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>All the properties required to asynchronously initiate a workflow. Use the <code>WorkflowTriggerTask</code> Object to create a task that initiates an instance of the specified workflow.</p> <p>The task is placed in the scheduling queue, and the workflow instance is initiated after the task reaches the top of the queue.</p> <p><code>task.WorkflowTriggerTask</code> does not successfully place a workflow job in queue if an identical instance of that workflow (with the same <code>recordType</code>, <code>id</code>, and <code>workflowId</code>) is currently executing or already in the scheduling queue.</p> <p>To use the <code>WorkflowTriggerTask</code> Object:</p> <ul style="list-style-type: none"> ■ Use <code>task.create(options)</code> to create the <code>WorkflowTriggerTask</code> Object. ■ Use <code>WorkflowTriggerTask.recordType</code> to set the record type of the workflow base record. ■ Use <code>WorkflowTriggerTask.recordId</code> to set the internal ID of the base record for the workflow. ■ Use <code>WorkflowTriggerTask.workflowId</code> to set the internal ID of the workflow that you want to run on the record specified by the <code>recordId</code>. ■ Optionally, use <code>WorkflowTriggerTask.params</code> to specify default values for workflow fields. ■ Use <code>WorkflowTriggerTask.submit()</code> to submit the asynchronous workflow initiation task to the NetSuite task queue. ■ Use the properties for the <code>WorkflowTriggerTaskStatus.status</code> object to get the status of the workflow execution. <p>Use the following guidelines with the <code>WorkflowTriggerTask</code> Object:</p> <ul style="list-style-type: none"> ■ <code>WorkflowTriggerTask.submit()</code> does not successfully place a workflow task in the scheduling queue if an identical instance of that workflow, with the same <code>recordType</code>, <code>recordId</code>, and <code>workflowId</code>, is currently executing or already in the scheduling queue. <p>For a complete list of this object's methods and properties, see WorkflowTriggerTask Object Members.</p> <p>To initiate a workflow on demand, see workflow.initiate(options).</p>
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/task Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var workflowTask = task.create({taskType: task.TaskType.WORKFLOW_TRIGGER});
4 workflowTask.recordType = 'customer';
5 workflowTask.recordId = 107;
6 workflowTask.workflowId = 3;
7 var taskId = workflowTask.submit();
8 ...
9 //Add additional code

```

WorkflowTriggerTask.submit()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Directs NetSuite to place the asynchronous workflow initiation task into the NetSuite scheduling queue and returns a unique ID for the task. Use WorkflowTriggerTaskStatus.status to view the status of a submitted task.
Returns	string The task ID
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
FAILED_TO_SUBMIT_JOB_REQUEST_1	Failed to submit job request: {reason}	Task cannot be submitted.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var workflowTriggerTask = workflowTask.submit();
4 ...
5 //Add additional code

```

WorkflowTriggerTask.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The ID of the task.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.WorkflowTriggerTask
Sibling Object Members	WorkflowTriggerTask Object Members
Since	

WorkflowTriggerTask.params

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Key/value pairs to set default values on fields specific to the workflow. These can include fields on the Workflow Definition Page or workflow and state Workflow Custom Fields .
Type	Object
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 task.params = {context: portlet}
4 ...
5 //Add additional code

```

WorkflowTriggerTask.recordId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID of the base record. For example, 55 or 124.
Type	number
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 workflowTask.recordId = 107;
4 ...
5 //Add additional code

```

WorkflowTriggerTask.recordType

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Record type of the workflow definition base record. For example, customer, salesorder, or lead. In the Workflow Manager, this is the record type that is specified in the Record Type field.
Type	string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 workflowTask.recordType = 'customer';
4 ...
5 //Add additional code

```

WorkflowTriggerTask.workflowId

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Internal ID (as a number), or script ID (as a string), for the workflow definition. This is the ID that appears in the ID field on the Workflow Definition Page .
Type	number string
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 workflowTask.workflowId = 3;
4 ...

```

```
5 | //Add additional code
```

task.WorkflowTriggerTaskStatus

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The status of an asynchronous workflow initiation task placed into the NetSuite task queue by WorkflowTriggerTask.submit() . Use task.checkStatus(options) with the unique ID for the asynchronous workflow initiation task to get the WorkflowTriggerTaskStatus object. For a complete list of this object's properties, see WorkflowTriggerTaskStatus Object Members .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```
1 //Add additional code
2 ...
3 var workflowTaskStatus = task.checkStatus(taskId);
4 if (workflowTaskStatus.status === task.TaskStatus.FAILED)
5 ...
6 //Add additional code
```

WorkflowTriggerTaskStatus.status

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Status for an asynchronous workflow placed in the NetSuite task queue by WorkflowTriggerTask.submit() . Returns a task.TaskStatus enum value.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Errors

Error Code	Message	Thrown If
READ_ONLY		Setting the property is attempted

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 var summary = task.checkStatus(scriptTaskId);
4 log.audit('Status', summary.status);
5 ...
6 ...
7 //Add additional code

```

WorkflowTriggerTaskStatus.taskId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The task ID associated with the specified task.
Type	string (read-only)
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Parent Object	task.WorkflowTriggerTaskStatus
Sibling Object Members	WorkflowTriggerTaskStatus Object Members
Since	

Errors

Error Code	Error Message	Thrown If
READ_ONLY		Setting the property is attempted.

task.checkStatus(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a task status object associated with a specific task ID. You can check the status of the following task types: <ul style="list-style-type: none">■ CSV import■ Entity deduplication■ Map/reduce script■ Query■ Record action■ Scheduled script■ Search■ SuiteQL
---------------------------	---

	<ul style="list-style-type: none"> ■ Workflow trigger
Returns	task.CsvImportTaskStatus task.EntityDeduplicationTaskStatus task.MapReduceScriptTaskStatus task.QueryTaskStatus task.RecordActionTaskStatus task.ScheduledScriptTaskStatus task.SearchTaskStatus task.SuiteQLTaskStatus task.WorkflowTriggerTaskStatus
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/task Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.taskId	string	required	Unique ID of the task to check the status of.

Syntax

 **Important:** The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var myTask = task.create({
4     taskType: task.TaskType.QUERY
5 });
6
7 // Set the properties of the myTask object
8
9 // Submit the task
10 var myTaskId = myTask.submit();
11
12 // Check the status
13 var myTaskStatus = task.checkStatus({
14     taskId: myTaskId
15 });
16 ...
17 // Add additional code

```

task.create(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a task object for the specified task type and returns the object.</p> <p>Use this method to create tasks to do the following:</p> <ul style="list-style-type: none"> ■ Schedule scripts ■ Run map/reduce scripts
---------------------------	---

	<ul style="list-style-type: none"> ■ Import CSV files ■ Merge duplicate records ■ Execute asynchronous searches, constructed queries, SuiteQL queries, and workflows
Returns	task.CsvImportTask task.EntityDeduplicationTask task.MapReduceScriptTask task.RecordActionTask task.QueryTask task.ScheduledScriptTask task.SearchTask task.SuiteQLTask task.WorkflowTriggerTask
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/task Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.taskType	string	required	<p>The type of task object to create.</p> <p>Use the task.TaskType enum to set the value.</p>
options.deploymentId	string	optional	<p>The script ID (as a string) of the script deployment record.</p> <p>This parameter sets the value for the ScheduledScriptTask.deploymentId or MapReduceScriptTask.deploymentId property.</p> <p>This parameter is applicable only when options.taskType is set to <code>task.TaskType.SCHEDULED_SCRIPT</code> or <code>task.TaskType.MAP_REDUCE</code>.</p>
options.params	Object Array<string boolean number>	optional	<p>An object that represents additional parameters and values for the task.</p> <p>This parameter accepts different values based on the value of the options.taskType parameter:</p> <ul style="list-style-type: none"> ■ If options.taskType is set to <code>task.TaskType.SCHEDULED_SCRIPT</code>, <code>task.TaskType.MAP_REDUCE</code> or <code>task.TaskType.WORKFLOW_TRIGGER</code>, this parameter accepts key-value pairs that override static script parameter field values on the script deployment record. For more information, see the help topic Creating Script Parameters Overview. For workflow tasks, keys can include fields on the Workflow Definition Page, or workflow and state custom fields. For more information, see the help topic Workflow Custom Fields. ■ If options.taskType is set to <code>task.TaskType.SUITE_QL</code>, this parameter accepts an array of parameters for the associated SuiteQL query. The specified parameters are substituted into the SuiteQL query string when you submit the task. <p>This parameter sets the value for the ScheduledScriptTask.params,</p>

Parameter	Type	Required / Optional	Description
			<p>MapReduceScriptTask.params, WorkflowTriggerTask.params, or SuiteQLTask.params properties.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.SCHEDULED_SCRIPT, task.TaskType.MAP_REDUCE, task.TaskType.WORKFLOW_TRIGGER, or task.TaskType.SUITE_QL.</p>
options.scriptId	number string	optional	<p>The internal ID (as a number) or script ID (as a string) for the script record.</p> <p>This parameter sets the value for the ScheduledScriptTask.scriptId or MapReduceScriptTask.scriptId property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.SCHEDULED_SCRIPT or task.TaskType.MAP_REDUCE.</p>
options.importFile	file.File string	optional	<p>A CSV file to import. Use a file.File object or a string that represents the CSV text to be imported.</p> <p>This parameter sets the value for the CsvImportTask.importFile property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.CSV_IMPORT.</p>
options.linkedFiles	Object	optional	<p>A map of key/value pairs that sets the data to be imported in a linked file for a multi-file import job, by referencing a file in the File Cabinet or the raw CSV data to import.</p> <p>The key is the internal ID of the record sublist for which data is being imported and the value is either a file.File object or the raw CSV data to import.</p> <p>You can assign multiple types of values to this property.</p> <p>This parameter sets the value for the CsvImportTask.linkedFiles property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.CSV_IMPORT.</p>
options.mappingId	number string	optional	<p>The internal ID (as a number) or script ID (as a string) of a saved import map created using the Import Assistant. See task.CsvImportTask.</p> <p>This parameter sets the value for the CsvImportTask.mappingId property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.CSV_IMPORT.</p>
options.name	string	optional	<p>The name for the CSV import task.</p> <p>You can optionally set a different name for a scripted import task. In the UI, this name appears on the CSV Import Job Status page.</p> <p>This parameter sets the value for the CsvImportTask.name property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.CSV_IMPORT.</p>
options.queueId	number	optional	Overrides the Queue Number property under Advanced Options on the Import Options page of the Import

Parameter	Type	Required / Optional	Description
			<p>Assistant. Use this property to programmatically select an import queue and improve performance during the import.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> Note: This property is only available if you have a SuiteCloud Plus license. For more information about using multiple queues when importing CSV files, see the help topics Queue Number and Use Multiple Threads and Multiple Queues to Run CSV Import Jobs. </div> <p>This parameter sets the value for the CsvImportTask.queueId property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.CSV_IMPORT.</p>
options.dedupeMode	string	optional	<p>Sets the mode for merging or deleting duplicate records.</p> <p>This parameter sets the value for the EntityDeduplicationTask.dedupeMode property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.ENTITY_DEDUPLICATION.</p> <p>Use the task.DedupeMode enum to set the value.</p>
options.entityType	string	optional	<p>Sets the type of entity on which you want to merge duplicate records.</p> <p>This parameter sets the value for the EntityDeduplicationTask.entityType property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.ENTITY_DEDUPLICATION.</p> <p>Use the task.DedupeEntityType enum to set the value.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> Note: If you specify a value of CUSTOMER for this parameter, NetSuite automatically includes prospects and leads in the task request. </div>
options.masterRecordId	number	optional	<p>When you merge duplicate records, you can delete all duplicates for a record or merge information from the duplicate records into the master record.</p> <p>Use this property to set the ID of the master record that you want to use as the master record in the merge.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> Important: You must also select SELECT_BY_ID for the EntityDeduplicationTask.masterSelectionMode property, or NetSuite ignores this setting. </div> <p>This parameter sets the value for the EntityDeduplicationTask.masterRecordId property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.ENTITY_DEDUPLICATION.</p>
options.masterSelectionMode	string	optional	When you merge duplicate records, you can delete all duplicates for a record or merge information from the duplicate records into the master record.

Parameter	Type	Required / Optional	Description
			<p>Set this property to determine which of the duplicate records to keep or to select the master record to use by ID.</p> <p>This parameter sets the value for the EntityDeduplicationTask.masterSelectionMode property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.ENTITY_DEDUPLICATION.</p> <p>Use the task.MasterSelectionMode enum to set the value.</p>
options.recordIds	number[]	optional	<p>The internal IDs of the records to perform the merge or delete operation on.</p> <p>You can use the search.duplicates(options) method to identify duplicate records or create an array with record internal IDs.</p> <p>This parameter sets the value for the EntityDeduplicationTask.recordIds property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.ENTITY_DEDUPLICATION.</p>
options.recordId	number	optional	<p>The internal ID of the base record.</p> <p>This parameter sets the value for the WorkflowTriggerTask.recordId property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.WORKFLOW_TRIGGER.</p>
options.recordType	string	optional	<p>The record type of the workflow definition base record, such as customer, salesorder, or lead.</p> <p>In the Workflow Manager, this is the record type that is specified in the Record Type field.</p> <p>This parameter sets the value for the WorkflowTriggerTask.recordType property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.WORKFLOW_TRIGGER.</p>
options.workflowId	number string	optional	<p>The internal ID (as a number) or script ID (as a string) for the workflow definition.</p> <p>This is the ID that appears in the ID field on the Workflow Definition Page.</p> <p>This parameter sets the value for the WorkflowTriggerTask.workflowId property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.WORKFLOW_TRIGGER.</p>
options fileId	number	optional	<p>The internal ID of the CSV file to export search results to. For more information about working with files, see N/file Module.</p> <p>This parameter is mutually exclusive with the options.filePath parameter. If you specify values for both parameters, an error occurs.</p> <p>This parameter sets the value for the SearchTask.fileId, QueryTask.fileId, or SuiteQLTask.fileId properties.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.SEARCH, task.TaskType.QUERY, or task.TaskType.SUITE_QL.</p>

Parameter	Type	Required / Optional	Description
options.filePath	string	optional	<p>The path of the CSV file to export search results to. For more information about working with files, see N/file Module.</p> <p>This parameter is mutually exclusive with the options.fileId parameter. If you specify values for both parameters, an error occurs.</p> <p>This parameter sets the value for the SearchTask.filePath, QueryTask.filePath, or SuiteQLTask.filePath properties.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.SEARCH, task.TaskType.QUERY, or task.TaskType.SUITE_QL.</p>
options.query	query.Query string	optional	<p>The query definition for a query task or SuiteQL task.</p> <p>This parameter accepts different values based on the value of the options.taskType parameter:</p> <ul style="list-style-type: none"> ▪ If options.taskType is set to task.TaskType.QUERY, this parameter accepts a query.Query object that represents the query to run. ▪ If options.taskType is set to task.TaskType.SUITE_QL, this parameter accepts a string that represents the SuiteQL query to run. <p>This parameter sets the value for the QueryTask.query or SuiteQLTask.query properties.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.QUERY or task.TaskType.SUITE_QL.</p>
options.savedSearchId	string	optional	<p>The internal ID of the saved search to be executed during the task.</p> <p>This parameter sets the value for the SearchTask.savedSearchId property.</p> <p>This parameter is applicable only when options.taskType is set to task.TaskType.SEARCH.</p>

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var mrTask = task.create({
4   taskType: task.TaskType.MAP_REDUCE,
5   scriptId: 34,
6   deploymentId: 'custdeploy1',
7   params: {
8     doSomething: true
9   }
10 });
11 ...
12 // Add additional code

```

task.ActionCondition



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Enumeration that holds the string values for the possible record action conditions.</p> <p>This enum is returned by RecordActionTask.condition.</p> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/task Module
Since	2019.1

Values

ALL_QUALIFIED_INSTANCES

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 // Add additional code
2 ...
3 var recordActionTask = task.create({taskType: task.TaskType.RECORD_ACTION});
4 recordActionTask.recordType = 'timebill';
5 recordActionTask.action = 'approve';
6 recordActionTask.condition = task.ActionCondition.ALL_QUALIFIED_INSTANCES;
7 recordActionTask.paramCallback = function(v) {
8   return { recordId: v, note: "this is a note for " + v };
9 };
10 var handle = recordActionTask.submit();
11 ...
12 // Add additional code

```

task.DedupeEntityType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Enumeration that holds the string values for entity types for which you can merge duplicate records with task.EntityDeduplicationTask.</p> <p>Use this enum for the EntityDeduplicationTask.entityType.</p>
-------------------------	--

	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Values

- CUSTOMER
- CONTACT
- VENDOR
- PARTNER
- LEAD
- PROSPECT

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [task Module Script Sample](#).

```

1 //Add additional code
2 ...
3 dedupeTask.entityType = task.DedupeEntityType.CUSTOMER;
4 ...
5 //Add additional code

```

task.DedupeMode

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Enumeration that holds the string values for the available deduplication modes when merging duplicate records with <code>task.EntityDeduplicationTask</code> . Use this enum for the <code>EntityDeduplicationTask.dedupeMode</code> property.
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>

Module	N/task Module
Since	2015.2

Values

- MERGE
- DELETE
- MAKE_MASTER_PARENT
- MARK_AS_NOT_DUPES

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 dedupeTask.dedupeMode = task.DedupeMode.MERGE;
4 ...
5 //Add additional code

```

task.MapReduceStage

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Enumeration that holds the string values for possible stages in task.MapReduceScriptTask for a map/reduce script. This enum is returned by MapReduceScriptTaskStatus.stage . For general information about map/reduce stages, see the help topics Map/Reduce Key Concepts and SuiteScript 2.0 Map/Reduce Script Stages .
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Since	2015.2

Values

- GET_INPUT

- MAP
- SHUFFLE
- REDUCE
- SUMMARIZE

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 if (summary.stage === task.MapReduceStage.SUMMARIZE)
4 ...
5 //Add additional code

```



Note: For general information about map/reduce scripts, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

task.MasterSelectionMode



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Enumeration that holds the string values for supported master selection modes when merging duplicate records with task.EntityDeduplicationTask.</p> <p>Use this enum for the EntityDeduplicationTask.masterSelectionMode property.</p> <p>For more information about these values, see the help topic Merging or Deleting Duplicate Records.</p> <div data-bbox="463 1275 1380 1417" style="border: 1px solid #ccc; padding: 10px;"> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p> </div>
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/task Module
Since	2015.2

Values

- CREATED_EARLIEST
- MOST_RECENT_ACTIVITY
- MOST_POPULATED_FIELDS
- SELECT_BY_ID

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 //Add additional code
2 ...
3 dedupeTask.masterSelectionMode = task.MasterSelectionMode.MOST_RECENT_ACTIVITY;
4 ...
5 //Add additional code

```

task.TaskStatus



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds the string values for possible task statuses.</p> <p>The following properties use values from this enum:</p> <ul style="list-style-type: none"> ▪ CsvImportTaskStatus.status ▪ EntityDeduplicationTaskStatus.status ▪ MapReduceScriptTaskStatus.status ▪ RecordActionTaskStatus.status ▪ QueryTaskStatus.status ▪ ScheduledScriptTaskStatus.status ▪ SearchTaskStatus.status ▪ SuiteQLTaskStatus.status ▪ WorkflowTriggerTaskStatus.status <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.</p>
Supported Script Types	<p>Server scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/task Module
Sibling Module Members	N/task Module Members
Since	2015.2

Values

Value
COMPLETE
FAILED
PENDING

Value
PROCESSING

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 if (status === task.TaskStatus.COMPLETE || status === task.TaskStatus.FAILED) {
4     // Handle the status
5 }
6 ...
7 // Add additional code

```

task.TaskType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for the types of task objects you can create using task.create(options) . Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value..
Supported Script Types	Server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task Module
Sibling Module Members	N/task Module Members
Since	2015.2

Values

Value
CSV_IMPORT
ENTITY_DEDUPLICATION
MAP_REDUCE
QUERY
RECORD_ACTION
SCHEDULED_SCRIPT

Value
SEARCH
SUITE_QL
WORKFLOW_TRIGGER

Syntax



Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see task Module Script Sample.

```

1 // Add additional code
2 ...
3 var mrTask = task.create({
4   taskType: task.TaskType.MAP_REDUCE
5 });
6 ...
7 // Add additional code

```

N/task/accounting/recognition Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/task/accounting/recognition module to merge revenue arrangements or revenue elements. A revenue arrangement is a transaction that records the details of a sale for the purposes of revenue allocation and recognition. The N/task/accounting/recognition module lets you combine revenue arrangements or revenue elements from multiple sources to represent a single contract obligation for revenue allocation and recognition.

You can use the [recognition.create\(options\)](#) method to create a merge task that combines entire revenue arrangements or individual revenue elements. This method returns a [recognition.MergeArrangementsTask](#) object (when merging revenue arrangements) or [recognition.MergeElementsTask](#) object (when merging revenue elements). After you obtain one of these objects, you can set its properties, such as the list of arrangements or elements to merge, the date on the merged revenue arrangement, whether to prospectively merge arrangements, and so on. You can use these properties to specify the same input data that you can specify when you merge revenue arrangements using the NetSuite UI. After you set its properties, you can submit the task for processing. Merge tasks are processed asynchronously.

You can use the [recognition.checkStatus\(options\)](#) method to check the status of a submitted merge task. This method returns a [recognition.MergeArrangementsTaskStatus](#) object that describes the current status of the merge task (pending, processing, complete, or failed). This object represents the current status for either a [recognition.MergeArrangementsTask](#) or a [recognition.MergeElementsTask](#). If the task completes successfully, this object includes the ID of the merged revenue arrangement record that was created. If the task fails, this object includes an error message that describes the failure.

To merge revenue arrangements or revenue elements using the N/task/accounting/recognition module, the following requirements must be met:

- The Advanced Revenue Management feature must be enabled in your account. For more information, see the help topic [Enabling the Advanced Revenue Management Feature](#).
- Your role must have the (Transactions) Revenue Arrangement permission assigned at a level of Create or higher. For more information, see the help topic [NetSuite Permissions Overview](#).

For more information about revenue arrangements, see the following help topics:

- [Revenue Arrangement Management](#): This topic describes revenue arrangements in general.
- [Combination and Modification of Performance Obligations](#): This topic describes the different types of merge results (combined revenue arrangements and prospective change orders).
- [Revenue Arrangement](#): This topic describes the revenue arrangement record type, including scripting considerations, supported script types, and sublist fields.

In this help topic

- [N/task/accounting/recognition Module Members](#)
- [MergeArrangementsTask Object Members](#)
- [MergeArrangementsTaskStatus Object Members](#)
- [MergeElementsTask Object Members](#)
- [N/task/accounting/recognition Module Script Samples](#)
 - [Sample 1: Merge revenue elements using internal IDs](#)
 - [Sample 2: Merge revenue arrangements using a saved search](#)
 - [Sample 3: Merge revenue arrangements using an ad-hoc search](#)

N/task/accounting/recognition Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	recognition.MergeArrangementsTask	Object	Server-side scripts	Encapsulates a task to merge all of the revenue elements from a specified list of revenue arrangements. Use recognition.create(options) to create this object.
	recognition.MergeArrangementsTaskStatus	Object	Server-side scripts	Encapsulates the current status of a submitted merge task. Use recognition.checkStatus(options) to create this object.
	recognition.MergeElementsTask	Object	Server-side scripts	Encapsulates a task to merge all of the specified revenue elements. Use recognition.create(options) to create this object.
Method	recognition.create(options)	recognition.MergeArrangementsTask recognition.MergeElementsTask	Server-side scripts	Creates a merge task that combines entire revenue arrangements or individual revenue elements. Use values in the recognition.TaskType enum to specify the type of merge task to create.
	recognition.checkStatus(options)	recognition.MergeArrangementsTaskStatus	Server-side scripts	Checks the status of a submitted merge task.
Enum	recognition.TaskStatus	enum	Server-side scripts	Holds the string values for supported merge task statuses. This enum is used to represent the task status in a recognition.MergeArrangementsTaskStatus object.
	recognition.TaskType	enum	Server-side scripts	Holds the string values for supported merge task types.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
				This enum is used to pass the task type argument to recognition.create(options) .

MergeArrangementsTask Object Members

The following members are called on the [recognition.MergeArrangementsTask](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	MergeArrangementsTask.submit()	number (read-only)	Server-side scripts	Submits the merge task for processing. This method returns a task ID that uniquely identifies the merge task.
Property	MergeArrangementsTask.arrangements	Array<number string> (read-only)	Server-side scripts	Holds an array of internal IDs of the revenue arrangement records to merge.
	MergeArrangementsTask.contractAcquisitionExpenseAccount	number string (read-only)	Server-side scripts	References the contract acquisition expense account for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . The default value is the account specified by the accounting preference Contract Acquisition Expense Account in your account.
	MergeArrangementsTask.contractAcquisitionDeferredExpenseAccount	number string (read-only)	Server-side scripts	References the contract acquisition deferred expense account for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . The default value is the account specified by the accounting preference Contract Acquisition Deferred Expense Account in your account.
	MergeArrangementsTask.contractCostAccrualDate	JavaScript Date (read-only)	Server-side scripts	Describes the contract cost accrual date to use for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . The default value is today's date.
	MergeArrangementsTask.mergeResidualRevenueAmounts	boolean (read-only)	Server-side scripts	Indicates whether the revenue arrangements are merged prospectively. For more information about prospective merges, see the help topic Prospective Merges . The default value is false.
	MergeArrangementsTask.recalculateResidualFairValue	boolean (read-only)	Server-side scripts	Indicates whether to recalculate the fair value on residual elements when revenue arrangements are prospectively merged. For more information about prospective merges, see the help topic Prospective Merges . This property can be set to true only if the MergeArrangementsTask.mergeResidualRevenueAmounts property is also set to true. The default value is false.
	MergeArrangementsTask.revenueArrangementDate	JavaScript Date (read-only)	Server-side scripts	Describes the date of the new revenue arrangement. The default value is today's date.

MergeArrangementsTaskStatus Object Members

The following members are called on the [recognition.MergeArrangementsTaskStatus](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	MergeArrangementsTaskStatus.errorMessage	string (read-only)	Server-side scripts	Holds an error message that describes the failure of the merge task. This property is valid only if the value of the status property is TaskStatus.FAILED.
	MergeArrangementsTaskStatus.inputArrangements	number[] (read-only)	Server-side scripts	Holds an array of internal IDs of the revenue arrangement records to merge. This property is valid only if the merge task was created using a task type of TaskType.MERGE_ARRANGEMENTS_TASK.
	MergeArrangementsTaskStatus.inputElements	number[] (read-only)	Server-side scripts	Holds an array of internal IDs of the revenue elements to merge. This property is valid only if the merge task was created using a task type of TaskType.MERGE_ELEMENTS_TASK.
	MergeArrangementsTaskStatus.resultingArrangement	number string (read-only)	Server-side scripts	References the internal ID of the new revenue arrangement that was created. This property is valid only if the value of the status property is TaskStatus.COMPLETE.
	MergeArrangementsTaskStatus.status	string (read-only)	Server-side scripts	Represents the current status of the merge task. This property uses values in the recognition.TaskType num.
	MergeArrangementsTaskStatus.submissionId	number string (read-only)	Server-side scripts	References the submission ID of the merge arrangements bulk process.
	MergeArrangementsTaskStatus.taskId	number string (read-only)	Server-side scripts	Holds the task ID of the merge task. The task ID is assigned to the merge task when you call recognition.create(options) .

MergeElementsTask Object Members

The following members are called on the [recognition.MergeElementsTask](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	MergeElementsTask.submit()	number (read-only)	Server-side scripts	Submits the merge task for processing. This method returns a task ID that uniquely identifies the merge task.
Property	MergeElementsTask.contractAcquisitionExpenseAccount	number string (read-only)	Server-side scripts	References the contract acquisition expense account for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . The default value is the account specified by the accounting preference Contract Acquisition Expense Account in your account.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	MergeElementsTask.contractAcquisitionDeferredExpenseAccount	number string (read-only)	Server-side scripts	<p>References the contract acquisition deferred expense account for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization.</p> <p>The default value is the account specified by the accounting preference Contract Acquisition Deferred Expense Account in your account.</p>
	MergeElementsTask.contractCostAccrualDate	JavaScript Date (read-only)	Server-side scripts	<p>Describes the contract cost accrual date to use for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization.</p> <p>The default value is today's date.</p>
	MergeElementsTask.elements	Array<number string> (read-only)	Server-side scripts	Holds an array of internal IDs of the revenue element records to merge.
	MergeElementsTask.revenueArrangementDate	JavaScript Date (read-only)	Server-side scripts	<p>Describes the date of the new revenue arrangement.</p> <p>The default value is today's date.</p>

N/task/accounting/recognition Module Script Samples

The following script samples demonstrate how to use the features of the N/task/accounting/recognition module.

Sample 1: Merge revenue elements using internal IDs

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample adds the internal IDs of several revenue element records to an array. It calls `recognition.create(options)` to create a merge task for revenue element records, uses the array as the list of revenue element records to merge, and submits the merge task. The sample also checks the status of the merge task.

If you run this sample code in your account, make sure to use the internal IDs of valid revenue element records in your account.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/task/accounting/recognition'], function(recognition){
6     var elementsList = [];
7     elementsList.push(401);
8     elementsList.push(402);
9
10    var recognitionTask = recognition.create({

```

```

11     taskType: recognition.TaskType.MERGE_ELEMENTS_TASK
12   });
13   recognitionTask.elements = elementsList;
14   var taskStatusId = recognitionTask.submit();
15
16   var mergeTaskState = recognition.checkStatus({
17     taskId: taskStatusId
18   });
19
20   log.debug('Submission ID = ' + mergeTaskState.submissionId);
21   log.debug('Resulting Arrangement ID = ' + mergeTaskState.resultingArrangement);
22   log.debug('status = ' + mergeTaskState.status);
23 });

```

Sample 2: Merge revenue arrangements using a saved search

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads a saved search for revenue arrangement records. It obtains the value of the internalid field from each record in the result set, and it adds the values to an array. It calls `recognition.create(options)` to create a merge task for revenue arrangement records, uses the array as the list of revenue arrangement records to merge, and submits the merge task. The sample also checks the status of the merge task and logs status information.

If you run this sample code in your account, make sure to use a saved search for valid revenue arrangement records in your account.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/task/accounting/recognition', 'N/search'], function(recognition, search){
6   var mySearch = search.load({
7     id: 'customsearch22'
8   });
9
10  var elementsList = [];
11  mySearch.run().each(function(result) {
12    var id = result.getValue({
13      name: 'internalid'
14    });
15    elementsList.push(id);
16  });
17
18  var recognitionTask = recognition.create({
19    taskType: recognition.TaskType.MERGE_ARRANGEMENTS_TASK
20  });
21
22  recognitionTask.arrangements = elementsList;
23  recognitionTask.revenueArrangementDate = new Date(2019, 2, 10);
24
25  var taskStatusId = recognitionTask.submit();
26  log.debug('taskId = ' + taskStatusId);
27
28  var mergeTaskState = recognition.checkStatus({
29    taskId: taskStatusId
30  });
31
32  log.debug('Submission ID = ' + mergeTaskState.submissionId);
33  log.debug('Resulting Arrangement ID = ' + mergeTaskState.resultingArrangement);
34  log.debug('status = ' + mergeTaskState.status);
35  log.debug('Error message = ' + mergeTaskState.errorMessage);
36 });

```

Sample 3: Merge revenue arrangements using an ad-hoc search

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates an ad-hoc search for revenue element records. It obtains the first 50 results, obtains the value of the elementsList field from each record in the result set, and adds the values to an array. It calls `recognition.create(options)` to create a merge task for revenue element records, uses the array as the list of revenue elements records to merge, and submits the merge task. This sample also checks the status of the merge task and logs status information.

```

1  /**
2  * @NApiVersion 2.x
3  */
4
5  require(['N/task/accounting/recognition', 'N/search'], function(recognition, search) {
6      var elementsList = [];
7      var rs = search.create({
8          type: 'revenueelement',
9          columns: [
10              'internalid'
11          ]
12      }).run();
13
14      var results = rs.getRange(0, 50);
15      for (var i = 0; i < results.length; i++) {
16          var id = result.getValue('elementsList');
17          elementsList.push(id);
18      }
19
20      var t = recognition.create({
21          taskType: recognition.TaskType.MERGE_ELEMENTS_TASK
22      });
23      t.elements = elementsList;
24      t.revenueArrangementDate = new Date(2019, 1, 1);
25
26      var taskId = t.submit();
27      log.debug('Initial status: ' + res.status);
28  });

```

recognition.MergeArrangementsTask

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a task to merge all of the revenue elements from a specified list of revenue arrangements. Use <code>recognition.create(options)</code> to create this object. After you create the object, you can populate its properties and submit the task for processing. The <code>MergeArrangementsTask.arrangements</code> property is required, and all other properties are optional.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task/accounting/recognition Module

Methods and Properties	MergeArrangementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTask.submit()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Submits the merge task for processing.</p> <p>This method returns a task ID that uniquely identifies the merge task. This task ID also represents the submission ID of the internal bulk process that performs the merge.</p> <p>Before you call this method to submit a merge task, you must populate the properties of the <code>recognition.MergeArrangementsTask</code> object (such as <code>MergeArrangementsTask.arrangements</code>, <code>MergeArrangementsTask.contractAcquisitionExpenseAccount</code>, and so on).</p>
Returns	number
Supported Script Types	<p>Server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	20 units
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTask
Sibling Object Members	MergeArrangementsTask Object Members
Since	2019.2

Errors

Error Code	Thrown If
NO_REVENUE_ARRANGEMENT_IDS_ARE_INCLUDED_IN_YOUR_INPUT	The <code>MergeArrangementsTask.arrangements</code> property is empty.
NO_REVENUE_ELEMENTS_WERE_FOUND_FOR_THE_REVENUERARRANGEMENT_IDS_YOU_INPUT	No revenue elements were found for the revenue arrangements specified in the <code>MergeArrangementsTask.arrangements</code> property.
WRONG_PARAMETER_TYPE	An invalid date format is specified in the <code>MergeArrangementsTask.contractCostAccrualDate</code> property or the <code>MergeArrangementsTask.revenueArrangementDate</code> property.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

1 | TBD

MergeArrangementsTask.arrangements



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Holds an array of internal IDs of the revenue arrangement records to merge. This property is required. You must specify a value for this property before you can submit the task for processing using MergeArrangementsTask.submit() . If you do not specify any revenue arrangement record IDs, a NO_REVENUER_ARRANGEMENT_IDS_ARE_INCLUDED_IN_YOUR_INPUT error is thrown when you call MergeArrangementsTask.submit() for the task. Invalid IDs are ignored. If no revenue elements were found for the specified revenue arrangement record IDs, a NO_REVENUER_ELEMENTS_WERE_FOUND_FOR_THE_REVENUER_ARRANGEMENT_IDS_YOU_INPUT error is thrown when you call MergeArrangementsTask.submit() .
Type	Array<number string>
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTask
Sibling Object Members	MergeArrangementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTask.contractAcquisitionExpenseAccount



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	References the contract acquisition expense account for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . If this preference is not enabled, this property is ignored. This property is optional. The default value is the account specified by the accounting preference Contract Acquisition Expense Account in your account.
Type	number string
Module	N/task/accounting/recognition Module

Parent Object	recognition.MergeArrangementsTask
Sibling Object Members	MergeArrangementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTask.contractAcquisitionDeferredExpenseAccount



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	References the contract acquisition deferred expense account for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . If this preference is not enabled, this property is ignored.
Type	number string
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTask
Sibling Object Members	MergeArrangementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTask.contractCostAccrualDate



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Describes the contract cost accrual date to use for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization
-----------------------------	---

	<p>is enabled. For more information, see the help topic Advanced Cost Amortization. If this preference is not enabled, this property is ignored.</p> <p>This property is optional. The default value is today's date. If you specify an invalid date format, a <code>WRONG_PARAMETER_TYPE</code> error is thrown when you call <code>MergeArrangementsTask.submit()</code> for the task.</p>
Type	JavaScript Date
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTask
Sibling Object Members	MergeArrangementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTask.mergeResidualRevenueAmounts



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the revenue arrangements are merged prospectively. For more information about prospective merges, see the help topic Prospective Merges . This property is optional. The default value is false.
Type	boolean
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTask
Sibling Object Members	MergeArrangementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTask.recalculateResidualFairValue



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether to recalculate the fair value on residual elements when revenue arrangements are prospectively merged. For more information about prospective merges, see the help topic Prospective Merges . This property is optional. This property can be set to true only if the MergeArrangementsTask.mergeResidualRevenueAmounts property is also set to true. If the MergeArrangementsTask.mergeResidualRevenueAmounts property is false, this property is ignored. The default value of this property is false.
Type	boolean
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTask
Sibling Object Members	MergeArrangementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTask.revenueArrangementDate



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Describes the date of the new revenue arrangement. This property is optional. The default value is today's date. If you specify an invalid date format, a WRONG_PARAMETER_TYPE error is thrown when you call MergeArrangementsTask.submit() for the task.
Type	JavaScript Date
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTask
Sibling Object Members	MergeArrangementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

recognition.MergeArrangementsTaskStatus



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the current status of a submitted merge task. Use recognition.checkStatus(options) to create this object. The current status corresponds to one of the values in the recognition.TaskStatus enum: PENDING, PROCESSING, COMPLETE, or FAILED.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task/accounting/recognition Module
Methods and Properties	MergeArrangementsTaskStatus Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTaskStatus.errorMessage



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Holds an error message that describes the failure of the merge task. This property is valid only if the value of the MergeArrangementsTaskStatus.status property is TaskStatus.FAILED.
Type	string (read-only)
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTaskStatus
Sibling Object Members	MergeArrangementsTaskStatus Object Members

Since	2019.2
-------	--------

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTaskStatus.inputArrangements



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Holds an array of internal IDs of the revenue arrangement records to merge. This property is valid only if the merge task was created using a task type of TaskType.MERGE_ARRANGEMENTS_TASK.
Type	number[] (read-only)
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTaskStatus
Sibling Object Members	MergeArrangementsTaskStatus Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTaskStatus.inputElements



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Holds an array of internal IDs of the revenue elements to merge. This property is valid only if the merge task was created using a task type of TaskType.MERGE_ELEMENTS_TASK.
Type	number[] (read-only)
Module	N/task/accounting/recognition Module

Parent Object	recognition.MergeArrangementsTaskStatus
Sibling Object Members	MergeArrangementsTaskStatus Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTaskStatus.resultingArrangement



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	References the internal ID of the new revenue arrangement that was created. This property is valid only if the value of the MergeArrangementsTaskStatus.status property is TaskStatus.COMPLETE.
Type	number (read-only)
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTaskStatus
Sibling Object Members	MergeArrangementsTaskStatus Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTaskStatus.status



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Represents the current status of the merge task. This property uses values in the recognition.TaskStatus enum.
Type	string (read-only)

Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTaskStatus
Sibling Object Members	MergeArrangementsTaskStatus Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTaskStatus.submissionId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	References the submission ID of the merge arrangements bulk process. This ID is the same as the task ID that is returned by MergeArrangementsTask.submit() or MergeElementsTask.submit() .
Type	number (read-only)
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTaskStatus
Sibling Object Members	MergeArrangementsTaskStatus Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeArrangementsTaskStatus.taskId



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Holds the task ID of the merge task. The task ID is assigned to the merge task when you call MergeArrangementsTask.submit() or MergeElementsTask.submit() for the task.
Type	number string (read-only)

Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeArrangementsTaskStatus
Sibling Object Members	MergeArrangementsTaskStatus Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

recognition.MergeElementsTask



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates a task to merge all of the specified revenue elements. Use recognition.create(options) to create this object. After you create the object, you can populate its properties and submit the task for processing. The MergeElementsTask.elements property is required, and all other properties are optional.
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/task/accounting/recognition Module
Methods and Properties	MergeElementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeElementsTask.submit()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Submits the merge task for processing.
---------------------------	--

	This method returns a task ID that uniquely identifies the merge task. This task ID also represents the submission ID of the internal bulk process that performs the merge.
Returns	number
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	20 units
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeElementsTask
Sibling Object Members	MergeElementsTask Object Members
Since	2019.2

Errors

Error Code	Thrown If
WRONG_PARAMETER_TYPE	An invalid date format is specified in the MergeElementsTask.contractCostAccrualDate property or the MergeElementsTask.revenueArrangementDate property.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/query Module Script Samples](#).

1 | TBD

MergeElementsTask.contractAcquisitionExpenseAccount

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	References the contract acquisition expense account for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . If this preference is not enabled, this property is ignored. This property is optional. The default value is the account specified by the accounting preference Contract Acquisition Expense Account in your account.
Type	number string
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeElementsTask
Sibling Object Members	MergeElementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeElementsTask.contractAcquisitionDeferredExpenseAccount



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	References the contract acquisition deferred expense account for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . If this preference is not enabled, this property is ignored. This property is optional. The default value is the account specified by the accounting preference Contract Acquisition Deferred Expense Account in your account.
Type	number string
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeElementsTask
Sibling Object Members	MergeElementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeElementsTask.contractCostAccrualDate



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Describes the contract cost accrual date to use for the new revenue arrangement. This property is valid only if the accounting preference Enable Advanced Cost Amortization is enabled. For more information, see the help topic Advanced Cost Amortization . This property is optional. The default value is today's date. If you specify an invalid date format, a WRONG_PARAMETER_TYPE error is thrown when you call MergeElementsTask.submit() for the task.
Type	JavaScript Date
Module	N/task/accounting/recognition Module

Parent Object	recognition.MergeElementsTask
Sibling Object Members	MergeElementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeElementsTask.elements



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Holds an array of internal IDs of the revenue element records to merge. This property is required. You must specify a value for this property before you can submit the task for processing using MergeElementsTask.submit() .
Type	Array<number string>
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeElementsTask
Sibling Object Members	MergeElementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

MergeElementsTask.revenueArrangementDate



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Describes the date of the new revenue arrangement. This property is optional. The default value is today's date. If you specify an invalid date format, a WRONG_PARAMETER_TYPE error is thrown when you call MergeElementsTask.submit() for the task.
-----------------------------	--

Type	JavaScript Date
Module	N/task/accounting/recognition Module
Parent Object	recognition.MergeElementsTask
Sibling Object Members	MergeElementsTask Object Members
Since	2019.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

```
recognition.create(options)
```



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a merge task that combines entire revenue arrangements or individual revenue elements. Use values in the recognition.TaskType enum to specify the type of merge task to create. After you call this method to create a merge task, you must specify the properties of the merge task (such as MergeArrangementsTask.arrangements or MergeElementsTask.elements) before you submit the task using MergeArrangementsTask.submit() or MergeElementsTask.submit() .
Returns	recognition.MergeArrangementsTask recognition.MergeElementsTask
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/task/accounting/recognition Module
Sibling Module Members	N/task/accounting/recognition Module Members
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.taskType	string	required	The type of merge task to create.

Parameter	Type	Required / Optional	Description
			Use values from the recognition.TaskType enum for this parameter.

Errors

Error Code	Thrown If
INVALID_TASK_TYPE	The options.taskType parameter represents an invalid task type.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

recognition.checkStatus(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Checks the status of a submitted merge task.
Returns	recognition.MergeArrangementsTaskStatus
Supported Script Types	Server-side scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	50 units
Module	N/task/accounting/recognition Module
Sibling Module Members	N/task/accounting/recognition Module Members
Since	2019.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.taskId	number string	required	The task ID of the merge task to check. The task ID is assigned to the merge task when you call recognition.create(options) .

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

recognition.TaskStatus



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds the string values for supported merge task statuses. This enum is used to represent the task status in a recognition.MergeArrangementsTaskStatus object.</p> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Type	enum
Module	N/task/accounting/recognition Module
Sibling Module Members	N/task/accounting/recognition Module Members
Since	2019.2

Values

Value
COMPLETE
FAILED
PENDING
PROCESSING

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

recognition.TaskType

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported merge task types. This enum is used to pass the task type argument to recognition.create(options) .
Type	enum
Module	N/task/accounting/recognition Module
Sibling Module Members	N/task/accounting/recognition Module Members
Since	2019.2

Values

Value
MERGE_ARRANGEMENTS_TASK
MERGE_ELEMENTS_TASK

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/task/accounting/recognition Module Script Samples](#).

1 | TBD

N/transaction Module

Note: The content in this help topic pertains to SuiteScript 2.0.

Load the transaction module to void transactions.

When you void a transaction, the total and all the line items for the transaction are set to zero. The transaction is not removed from the system. NetSuite supports two types of voids: direct voids and voids by reversing journal. For additional information, see the help topic [Voiding, Deleting, or Closing Transactions](#).

The type of void performed with your script depends on the targeted account's preference settings:

- If the Using Reversing Journals preference is **disabled**, a **direct void** is performed.
- If the Using Reversing Journals preference is **enabled**, a **void by reversing journal** is performed.



Important: After you successfully void a transaction, you can no longer make changes to the transaction that impact the general ledger.

- N/transaction Module Members
- N/transaction Module Script Samples

N/transaction Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	transaction.void(options)	number	Client and server-side scripts	Voids a transaction record.
	transaction.void.promise(options)	number	Client scripts	Voids a transaction record asynchronously.
Enum	transaction.Type	enum	Client and server-side scripts	Enumeration that holds the string values for supported record types.

N/transaction Module Script Samples

The following script samples demonstrate how to use the features of the N/transaction module.

Sample 1: Void a transaction



Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a sales order record, saves it, then voids the sales order. Before creating the sales order record, the sample loads a set of accounting preferences from the current NetSuite account, specifies that the REVERSALVOIDING preference should be disabled (set to false), and saves the preferences. This sample works only in NetSuite OneWorld accounts. Make sure to replace hard-coded values (such as record IDs) with valid values from your NetSuite account.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/transaction', 'N/config', 'N/record'], function(transaction, config, record) {
6   function voidSalesOrder() {
7     var accountingConfig = config.load({
8       type: config.Type.ACCTOUNG_PREFERENCES
9     });
10    accountingConfig.setValue({
11      fieldId: 'REVERSALVOIDING',
12      value: false
13    });
14    accountingConfig.save();
15 }

```

```

16
17     var salesOrderObj = record.create({
18         type: 'salesorder',
19         isDynamic: false
20     });
21     salesOrderObj.setValue({
22         fieldId: 'entity',
23         value: 107
24     });
25     salesOrderObj.setSublistValue({
26         sublistId: 'item',
27         fieldId: 'item',
28         value: 233,
29         line: 0
30     });
31     salesOrderObj.setSublistValue({
32         sublistId: 'item',
33         fieldId: 'amount',
34         value: 1,
35         line: 0
36     });
37
38     var salesOrderId = salesOrderObj.save();
39
40     var voidSalesOrderId = transaction void({
41         type: record.Type.SALES_ORDER,
42         id: salesOrderId
43     });
44
45     var salesOrder = record.load({
46         type: 'salesorder',
47         id: voidSalesOrderId
48     });
49
50     // The value of the memo field is 'VOID'
51     var memo = salesOrder.getValue({
52         fieldId: 'memo'
53     });
54 }
55
56 voidSalesOrder();
57 });

```

transaction void(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Method used to void a transaction record object and return an id that indicates the type of void performed.</p> <p>The type of void performed depends on the targeted account's preference settings.</p> <div style="background-color: #ffffcc; border: 1px solid #ffcc00; padding: 5px;"> ⚠ Important: After you void a transaction, you cannot make changes to the transaction that impact the general ledger. </div>
Returns	<p>An ID returned as a number.</p> <ul style="list-style-type: none"> ▪ If a direct void is performed, returns the ID of the record voided. ▪ If a void by reversing journal is performed, returns the ID of the newly created voiding journal.
Supported Script Types	<p>All client and server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>

Governance	10 units
Module	N/transaction Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.id	number string	required	Internal ID of the specific transaction record instance to void.	2015.2
options.type	string	required	Internal ID of the type of transaction record to void	2015.2

Errors

Error Code	Message	Thrown If
INVALID_RECORD_TYPE		The type argument passed is not valid or the record type is not voidable.
THAT_RECORD_DOES_NOT_EXIST		The id argument passed is not valid.
SSS_MISSING_REQD_ARGUMENT		The type or id argument is missing.

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see transaction Module Script Sample.

```

1 //Add additional code
2 ...
3 var voidSalesOrderId = transaction void({
4   type: transaction.Type.SALES_ORDER,
5   id: salesOrderId
6 });
7 ...
8 //Add additional code

```

transaction_void.promise(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Method used to void a transaction record object asynchronously and return an ID that indicates the type of void performed: <ul style="list-style-type: none"> ■ If a direct void is performed, this method returns the ID of the record that was voided. ■ If a void by reversing journal is performed, this method returns the ID of the newly created voiding journal.
---------------------------	--

	<p>The type of void performed depends on the targeted account's preference settings.</p> <p>Important: After you void a transaction, you cannot make changes to the transaction that impact the general ledger.</p> <p>Note: The parameters and errors thrown for this method are the same as those for transaction.void(options). For more information on promises, see Promise Object.</p>
Returns	Promise Object
Synchronous Version	transaction.void(options)
Supported Script Types	<p>All client-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Client Script Type.</p>
Governance	10 units
Module	N/transaction Module
Since	2015.2

Syntax

Important: The following code sample shows the syntax for this member. It is not a functional example. For a complete promise script example, see [Promise Object](#).

```

1 //Add additional code
2 ...
3 var voidSalesOrderId = transaction.void.promise({
4     type: record.Type.SALES_ORDER,
5     id: salesOrderId
6 });
7 ...
8 //Add additional code

```

transaction.Type

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Enumeration that holds the string values for supported transaction record types.
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	<p>All client and server-side scripts</p> <p>For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/transaction Module
Since	2015.2

Values

Transaction Record	Supported Void Type
ASSEMBLY_BUILD	None
ASSEMBLY_UNBUILD	None
BIN_TRANSFER	None
BIN_WORKSHEET	None
BLANKET_PURCHASE_ORDER	None
CASH_REFUND	Direct Void
CASH_SALE	Direct Void
CHECK	Void by Reversing Journal
CREDIT_CARD_CHARGE	None
CREDIT_CARD_REFUND	None
CREDIT_MEMO	Direct Void
CUSTOM_PURCHASE	None
CUSTOM_SALE	None
CUSTOMER_DEPOSIT	Direct Void
CUSTOMER_PAYMENT	Direct Void
CUSTOMER_PAYMENT_AUTHORIZATION	None
CUSTOMER_REFUND	Direct Void and Void by Reversing Journal
CUSTOM_TRANSACTION	None
DEPOSIT	None
DEPOSIT_APPLICATION	None
ESTIMATE	Direct Void
EXPENSE_REPORT	Direct Void
FULFILLMENT_REQUEST	None
INBOUND_SHIPMENT	None
INVENTORY_ADJUSTMENT	None
INVENTORY_COST_REVALUATION	None
INVENTORY_COUNT	None
INVENTORY_STATUS_CHANGE	None
INVENTORY_TRANSFER	None
INVOICE	Direct Void
ITEM_FULFILLMENT	None

Transaction Record	Supported Void Type
ITEM_RECEIPT	None
JOURNAL_ENTRY	Direct Void
OPPORTUNITY	None
PAYCHECK	None
PAYCHECK_JOURNAL	Direct Void
PERIOD_END_JOURNAL	None
PURCHASE_CONTRACT	None
PURCHASE_ORDER	None
PURCHASE_REQUSITION	None
RETURN_AUTHORIZATION	Direct Void
REVENUE_ARRANGEMENT	None
REVENUE_COMMITMENT	None
REVENUE_COMMITMENT_REVERSAL	None
SALES_ORDER	Direct Void
STORE_PICKUP_FULFILLMENT	None
TRANSFER_ORDER	Direct Void
VENDOR_BILL	Direct Void
VENDOR_CREDIT	Direct Void
VENDOR_PAYMENT	Direct Void and Void by Reversing Journal
VENDOR_PREPAYMENT	Direct Void and Void by Reversing Journal
VENDOR_PREPAYMENT_APPLICATION	Direct Void and Void by Reversing Journal
VENDOR_RETURN_AUTHORIZATION	Direct Void
WAVE	Direct Void
WORK_ORDER	Direct Void
WORK_ORDER_CLOSE	Direct Void
WORK_ORDER_COMPLETION	Direct Void
WORK_ORDER_ISSUE	Direct Void

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see record Module Script Samples.

```

1 //Add additional code
2 ...

```

```

3 | var voidSalesOrderId = transaction.void({
4 |   type: transaction.Type.SALES_ORDER,
5 |   id: salesOrderId
6 | });
7 | ...
8 | //Add additional code

```

N/translation Module



Note: The content in this help topic pertains to SuiteScript 2.0.

The N/translation module lets SuiteScript developers interact with NetSuite Translation Collections programmatically. For more information about Translation Collections, see the help topic [Translation Collections Overview](#).

You can watch a video that demonstrates how to use the N/translation module to work with Translation Collections.

Using the N/translation Module for Translation Collections

A Translation Collection is a customization object that stores translation strings with their translations. In 2019.1, a single Translation Collection can contain up to 1,000 translation strings. A translation string is a key/value pair where the key is an identifier and its value is a source string. A key references one string that can be translated into multiple languages. For example, a translation string for the word "hello" could consist of a key called HELLO and a string value of "hello". You can translate a string into any language supported by NetSuite. For a list of these languages, see the help topic [Configuring Multiple Languages](#).

You can create a collection of terms for translation in the NetSuite UI. To create this collection, your role must have the Manage Translations permission, or you must be using an Administrator role. You can export the collection of terms as an XLIFF translation file with a .xlf extension and send this file to a translation vendor. After the translation vendor translates the collection of terms, you can import the translation file back into your NetSuite account. You can use the collection of terms to translate labels and messages in your scripts, as well as in Suitelets and SuiteApps. For information about managing Translation Collections in the UI, see the help topic [About the Manage Translations Page](#).

You can use the N/translation module to access the translation strings stored in Translation Collections. The N/translation module provides read-only access to Translation Collections. Translation Collections are managed in the NetSuite UI, and you cannot create or modify Translation Collections using SuiteScript.

A Translation Collection is encapsulated in the [translation.Handle](#) object. The [translation.Handle](#) object is a hierarchical object, which means that each node in the object is either another [translation.Handle](#) object or a [translation.Translator](#) function. Translator functions combine strings with parameters. When you create a Translation Collection in the NetSuite UI, you can include parameter placeholders in your translation strings. The translator function injects the specified parameter values into the placeholders in the returned translation string.

In your scripts, use [translation.get\(options\)](#) to get a [translation.Translator](#) function you can use to obtain specific translated strings in a collection. Consider the following code sample:

```

1 | // key HELLO_1 = 'Hello, {1}'
2 |
3 | message: translation.get({
4 |   collection: 'custcollection_my_strings',
5 |   key: 'HELLO_1'
6 | })({
7 |   params: ['NetSuite']
8 | })

```

In this sample, if the string value of the HELLO_1 key is "Hello, {1}", the translation.Translator function combines the string with the params parameter value and returns "Hello, NetSuite". You can also use [translation.load\(options\)](#) to load translation strings from one or more Translation Collections. For information about the way strings are added to and formatted in collections, see the help topic [Working with Translation Collection Strings](#).

You can load collections in different language locales by using the locales parameter of [translation.load\(options\)](#). You can also use [translation.selectLocale\(options\)](#) to create a translation.Handle object in a specific locale from an existing translation.Handle object.

- [N/translation Module Members](#)
- [N/translation Module Script Samples](#)

N/translation Module Members

Member Type	Name	Return Type / Value Type	Supported Script Type	Description
Object	translation.Handle	Object	Client and server-side scripts	Encapsulates a Translation Collection for a locale.
	translation.Translator	Object / Function	Client and server-side scripts	Represents a translator function that returns translated strings. The translated strings include variables that are passed as parameters to the translator function.
Method	translation.get(options)	translation.Translator	Client and server-side scripts	Creates a translator function for a key in the specified Translation Collection and locale.
	translation.load(options)	translation.Handle	Client and server-side scripts	Creates a translation.Handle object with translations for the specified Translation Collections and locales.
	translation.selectLocale(options)	translation.Handle	Client and server-side scripts	Creates a translation.Handle object in the specified locale from an existing translation.Handle object.
Enum	translation.Locale	enum	Client and server-side scripts	Holds the supported locales for Translation Collections. This enum is used to pass the locale argument to translation.get(options) and translation.selectLocale(options) .

N/translation Module Script Samples

See the following script samples for examples of how to use the N/translation module.

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample accesses translation strings one at a time using [translation.get\(options\)](#). This method returns a translator function, which is subsequently called with any specified parameters. The translator function returns the string in the user's session locale by default.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/ui/message', 'N/translation'],
6         function(message, translation) {
7
8      // Create a message with translated strings
9      var myMsg = message.create({
10        title: translation.get({
11          collection: 'custcollection_my_strings',
12          key: 'MY_TITLE'
13        })(),
14        message: translation.get({
15          collection: 'custcollection_my_strings',
16          key: 'MY_MESSAGE'
17        })(),
18        type: message.Type.CONFIRMATION
19      });
20
21      // Show the message for 5 seconds
22      myMsg.show({
23        duration: 5000
24      });
25    });

```



Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample accesses translation strings using a locale other than the default locale. When you call `translation.get(options)` and do not specify a locale, the method uses the current user's session locale. You can use the `options.locale` parameter to specify another locale. The `translation.Locale` enum lists all locales that are enabled for a company, and you can use these locales in `translation.get(options)`. The `translation.Locale` enum also includes two special values: `CURRENT` and `COMPANY_DEFAULT`. The `CURRENT` value represents the current user's locale, and the `COMPANY_DEFAULT` value represents the default locale for the company.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/ui/message', 'N/translation'],
6         function(message, translation) {
7
8      // Create a message with translated strings
9      var myMsg = message.create({
10        title: translation.get({
11          collection: 'custcollection_my_strings',
12          key: 'MY_TITLE',
13          locale: translation.Locale.COMPANY_DEFAULT
14        })(),
15        message: translation.get({
16          collection: 'custcollection_my_strings',
17          key: 'MY_MESSAGE',
18          locale: translation.Locale.COMPANY_DEFAULT
19        })(),
20        type: message.Type.CONFIRMATION
21      });
22
23      // Show the message for 5 seconds
24      myMsg.show({
25        duration: 5000
26      });
27    });

```



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample accesses parametrized translation strings. When you create a Translation Collection in the NetSuite UI, you can include parameter placeholders in your translation strings. Placeholders use braces and a number (starting from 1). The translator function injects the specified parameter values into the placeholders in the translation string. For example, "Hello, {1}!" is a valid translation string, where {1} is a placeholder for a parameter. In this sample, the parameter "NetSuite" is provided to the translator function returned from [translation.get\(options\)](#), and the translator function returns a translated string of "Hello, NetSuite!"

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/ui/message', 'N/translation'],
6         function(message, translation) {
7
8      // Create a message with translated strings
9      var myMsg = message.create({
10        title: translation.get({
11          collection: 'custcollection_my_strings',
12          key: 'MY_TITLE'
13        }),
14        message: translation.get({
15          collection: 'custcollection_my_strings',
16          key: 'HELLO_1'
17        }),
18        params: ['NetSuite']
19      },
20      type: message.Type.CONFIRMATION
21    );
22
23    // Show the message for 5 seconds
24    myMsg.show({
25      duration: 5000
26    });
27  });

```



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads specific translation strings from a collection. The [translation.load\(options\)](#) method can load a maximum of 1,000 translation strings. If you need only a few of the translation strings in a collection, you can load only the strings you need instead of loading the entire collection.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/ui/message', 'N/translation'],
6         function(message, translation) {
7
8      // Load translation strings by key
9      var localizedStrings = translation.load({
10        collections: [
11          {
12            alias: 'myCollection',
13            collection: 'custcollection_my_strings',
14            keys: ['MY_TITLE', 'MY_MESSAGE']
15          }
16        ]
17      });

```

```

15    });
16
17    // Create a message with translated strings
18    var myMsg = message.create({
19        title: localizedStrings.myCollection.MY_TITLE(),
20        message: localizedStrings.myCollection.MY_MESSAGE(),
21        type: message.Type.CONFIRMATION
22    });
23
24    // Show the message for 5 seconds
25    myMsg.show({
26        duration: 5000
27    });
28 });

```

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads translation strings by key from multiple Translation Collections in a single call of [translation.load\(options\)](#). This method can load a maximum of 1,000 translation strings, regardless of whether the strings are loaded from one collection or multiple collections.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/ui/message', 'N/translation'],
6     function(message, translation) {
7
8     // Load two Translation Collections
9     var localizedStrings = translation.load({
10         collections: [
11             {
12                 alias: 'myCollection',
13                 collection: 'custcollection_my_strings',
14                 keys: ['MY_TITLE']
15             },
16             {
17                 alias: 'myOtherCollection',
18                 collection: 'custcollection_other_strings',
19                 keys: ['MY_OTHER_MESSAGE']
20             }
21         ]
22     });
23
24     // Create a message with translated strings
25     var myMsg = message.create({
26         title: localizedStrings.myCollection.MY_TITLE(),
27         message: localizedStrings.myOtherCollection.MY_OTHER_MESSAGE(),
28         type: message.Type.CONFIRMATION
29     });
30
31     // Show the message for 5 seconds
32     myMsg.show({
33         duration: 5000
34     });
35 });

```

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads translation strings by key from a Translation Collection with multiple locales. When you load translation strings using [translation.load\(options\)](#), you can specify a list of valid locales for the strings. You can use these locales when you select a locale using [translation.selectLocale\(options\)](#).

If you specify more than one locale when you call `translation.load(options)`, the first specified locale in the list is used for the created `translation.Handle` object. If you want to use a different locale from the list, use `translation.selectLocale(options)`, which returns a `translation.Handle` object in the specified locale. You must load a locale using `translation.load(options)` before you can select it using `translation.selectLocale(options)`.

```

1  /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/ui/message', 'N/translation'],
6   function(message, translation) {
7
8     // Load a Translation Collection and a set of locales
9     var germanStrings = translation.load({
10       collections: [
11         {
12           alias: 'myCollection',
13           collection: 'custcollection_my_strings',
14           keys: ['MY_TITLE', 'MY_MESSAGE']
15         },
16         locales: [translation.Locale.de_DE, translation.Locale.es_ES]
17     });
18
19     // Select a locale from the list of loaded locales
20     var spanishStrings = translation.selectLocale({
21       handle: germanStrings,
22       locale: translation.Locale.es_ES
23     });
24
25     // Create a message with translated strings
26     var myMsg = message.create({
27       title: germanStrings.myCollection.MY_TITLE(),
28       message: spanishStrings.myCollection.MY_MESSAGE(),
29       type: message.Type.CONFIRMATION
30     });
31
32     // Show the message for 5 seconds
33     myMsg.show({
34       duration: 5000
35     });
36 });

```

translation.Handle



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>Encapsulates a Translation Collection for a locale.</p> <p>Use <code>translation.load(options)</code> to create a <code>translation.Handle</code> object with translations for the specified Translation Collections and locales. Use <code>translation.selectLocale(options)</code> to create a <code>translation.Handle</code> object in the specified locale from an existing <code>translation.Handle</code> object.</p> <p>The <code>translation.Handle</code> object is a hierarchical object, which means that each node in the object is either another <code>translation.Handle</code> object or a <code>translation.Translator</code> function. Translator functions combine strings with parameters. When you create a Translation Collection in the NetSuite UI, you can include parameter placeholders in your translation strings. The translator function injects the specified parameter values into the placeholders in the returned translation string.</p>
Supported Script Types	<p>Client and server-side scripts</p> <p>For additional information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/translation Module

Sibling Object Members	N/translation Module Members
Since	2019.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/translation Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var localizedStrings = translation.load({
4   collections: [
5     {
6       alias: 'myCollection',
7       collection: 'custcollection_my_strings',
8       keys: ['MY_TITLE', 'MY_MESSAGE']
9     }
10 });
11 var myMsg = message.create({
12   title: localizedStrings.myCollection.MY_TITLE(),
13   message: localizedStrings.myCollection.MY_MESSAGE(),
14   type: message.Type.CONFIRMATION
15 });
16 ...
17 // Add additional code

```

```

1 /**
2  * @NApiVersion 2.0
3  * @NScriptType clientscript
4 */
5 define(['N/translation'], function(translation) {
6   return {
7     pageInit: function(context) {
8       var handle = translation.load({
9         collections: [
10           {
11             alias: "phrases", collection: "CUSTCOLLECTION_PHRASES",
12             keys: ["HELLO"]},
13           {
14             alias: "specialstrings", collection: "CUSTCOLLECTION_SPECIALSTRINGS",
15             keys: ["HELLO_1"]}
16         ],
17         locales: ["fr_FR", 'en_US' ] });
18       console.log(handle.phrases.HELLO());
19       //logs hello in company default language - Hello (if default is English)
20       var frenchHandle = translation.selectLocale({handle: result, locale: "fr_FR"});
21       console.log(frenchHandle.phrases.HELLO());
22       //logs hello in french - Bonjour
23     }
24   };
25 ...
26 // Add additional code

```

translation.Translator



Note: The content in this help topic pertains to SuiteScript 2.0.

Object / Function Description	Represents a translator function that returns translated strings.
--------------------------------------	---

	<p>Use translation.get(options) to obtain this function for the specified Translation Collection and locale. The translator function is called with any parameters that you specify, and the translator function returns the appropriate translated string.</p> <p>When you create a Translation Collection in the NetSuite UI, you can include parameter placeholders in your translation strings. Translation strings that include placeholders are called parametrized translation strings. Placeholders use braces and a number (starting from 1). The translator function injects the specified parameter values into the placeholders in the translation string.</p> <p>For example, "Hello, {1}!" is a valid translation string, where {1} is a placeholder for a parameter. If you call translation.get(options) and specify a parameter of "NetSuite", the translator function returns "Hello, NetSuite!" in the appropriate locale.</p>
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/translation Module
Sibling Object Members	N/translation Module Members
Since	2019.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.params	string[]	optional	The parameters to pass to the translator function. The parameter values are used in parametrized translation strings.

Errors

Error Code	Thrown If
WRONG_PARAMETER_TYPE	The function parameters were not passed as an array.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/translation Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myMsg = message.create({
4   title: translation.get({
5     collection: 'custcollection_my_strings',
6     key: 'MY_TITLE'
7   })(),
8   message: translation.get({
9     collection: 'custcollection_my_strings',
10    key: 'HELLO_1'
11  })(
12    params: ['NetSuite']
13  ),
14  type: message.Type.CONFIRMATION

```

```

15 });
16 ...
17 // Add additional code

```

translation.get(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a translator function for a key in the specified Translation Collection and locale. This method returns a translator function, which is subsequently called with any specified parameters. When you call <code>translation.get(options)</code> and do not specify a locale, the method uses the current user's session locale. You can use the <code>options.locale</code> parameter to specify another locale. The translation.Locale enum lists all locales that are enabled for a company, and you can use these locales in <code>translation.get(options)</code> .
Returns	<code>translation.Translator</code>
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/translation Module
Sibling Object Members	N/translation Module Members
Since	2019.1

Parameters



Note: The `options` parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.collection</code>	string	required	The script ID of the collection.
<code>options.key</code>	string	required	A valid key from the collection.
<code>options.locale</code>	string	optional	A valid locale from the translation.Locale enum. If a locale is not specified, the locale from the current session is used as the default locale.

Errors

Error Code	Thrown If
<code>SSS_MISSING_REQD_ARGUMENT</code>	A collection or key parameter is missing.
<code>INVALID_TRANSLATION_KEY</code>	The format of a specified key is invalid.
<code>INVALID_TRANSLATION_COLLECTION</code>	The format of a specified collection is invalid.
<code>INVALID_LOCALE</code>	The format of a specified locale is invalid.

Error Code	Thrown If
TRANSLATION_KEY_NOT_FOUND	A specified translation key was not found.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/translation Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myMsg = message.create({
4     title: translation.get({
5         collection: 'custcollection_my_strings',
6         key: 'MY_TITLE'
7    ())),
8     message: translation.get({
9         collection: 'custcollection_my_strings',
10        key: 'HELLO_1'
11   ())),
12    params: ['NetSuite']
13 },
14 type: message.Type.CONFIRMATION
15 );
16 ...
17 // Add additional code

```

translation.load(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Creates a translation.Handle object with translations for the specified Translation Collections and locales.</p> <p>This method returns a translation.Handle object with translation strings organized by collection and ID. Every node in a translation.Handle object is either another translation.Handle object or a translation.Translator function.</p> <p>You can load translation strings from multiple Translation Collections in a single call of translation.load(options). You must specify the keys of individual translation strings that you want to load. You cannot load all of the terms in a Translation Collection at one time. The translation.load(options) method can load a maximum of 1,000 translation strings, regardless of whether the strings are loaded from one collection or multiple collections.</p> <p>When you load translation strings using translation.load(options), you can specify a list of valid locales for the strings. You can use these locales when you select a locale using translation.selectLocale(options). If you specify more than one locale when you call translation.load(options), the first specified locale in the list is used for the created translation.Handle object. If you want to use a different locale from the list, use translation.selectLocale(options), which returns a translation.Handle object in the specified locale. You must load a locale using translation.load(options) before you can select it using translation.selectLocale(options).</p>
Returns	translation.Handle
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None

Module	N/translation Module
Sibling Object Members	N/translation Module Members
Since	2019.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.collections	Object[]	required	A list of translation.Handle objects to load.
options.collections.alias	string	required	An alias to identify the collection. This alias is used by the script to determine the collection to load.
options.collections.collection	string	required	The script ID of the collection to load.
options.collections.keys	string[]	required	A list of translation keys from the collection to load.
options.locales	string[]	optional	A list of locales to load the collection in. Use the values in the translation.Locale enum to set this value.

Errors

Error Code	Thrown If
WRONG_PARAMETER_TYPE	One of the array parameters (options.collections, options.collections.keys, or options.locales) is not an array.
SSS_MISSING_REQD_ARGUMENT	A collection or key parameter is missing.
INVALID_TRANSLATION_KEY	The format of a specified key is invalid.
INVALID_TRANSLATION_COLLECTION	The format of a specified collection is invalid.
INVALID_LOCALE	The format of a specified locale is invalid.
INVALID_ALIAS	The format of a specified alias is invalid.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/translation Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var localizedStrings = translation.load({
4   collections: [
5     {
        alias: 'myCollection',

```

```

6   collection: 'custcollection_my_strings',
7   keys: ['MY_TITLE', 'MY_MESSAGE']
8 });
9 });
10
11 var myMsg = message.create({
12   title: localizedStrings.myCollection.MY_TITLE()
13   message: localizedStrings.myCollection.MY_MESSAGE(),
14   type: message.Type.CONFIRMATION
15 });
16 ...
17 // Add additional code

```

translation.selectLocale(options)

Method Description	Creates a translation.Handle object in the specified locale from an existing translation.Handle object. This method returns a translation.Handle object that contains the same translation strings as the options.handle object, and the strings are in the options.locale locale. Before you can use this method to select a locale, the locale must be loaded using the locales parameter of translation.load(options) .
Returns	translation.Handle
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/translation Module
Sibling Object Members	N/translation Module Members
Since	2019.1

Parameters

 **Note:** The [options](#) parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.handle	translation.Handle	required	The translation.Handle object to select a locale for.
options.locale	string	required	The locale to select. Use the values in the translation.Locale enum to set this value.

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A handle or locale parameter is missing.

Error Code	Thrown If
WRONG_PARAMETER_TYPE	The options.handle parameter is not a translation.Handle object.
INVALID_LOCALE	The specified translation.Handle object uses an unknown or unsupported locale.
TRANSLATION_HANDLE_IS_IN_AN_ILLEGAL_STATE	The specified translation.Handle object is in an illegal state.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/translation Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var germanStrings = translation.load({
4   collections: [
5     {
6       alias: 'myCollection',
7       collection: 'custcollection_my_strings',
8       keys: ['MY_TITLE', 'MY_MESSAGE'],
9     }
10  ],
11  locales: [translation.Locale.de_DE, translation.Locale.es_ES]
12 });
13
14 var spanishStrings = translation.selectLocale({
15   handle: germanStrings,
16   locale: translation.Locale.es_ES
17 });
18 ...
19 // Add additional code

```

translation.Locale

 **Note:** JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Holds the string values for supported locales for Translation Collections. This enum is used to pass the locale argument to translation.get(options) , translation.selectLocale(options) , and translation.load(options) . This enum lists all locales that are enabled for a company. This enum also includes two special values: CURRENT and COMPANY_DEFAULT. The CURRENT value represents the current user's locale, and the COMPANY_DEFAULT value represents the default locale for the company.
Type	enum
Supported Script Types	Client and server-side scripts For additional information, see the help topic SuiteScript 2.0 Script Types .
Module	N/translation Module
Sibling Object Members	N/translation Module Members
Since	2019.1

Values

The following table lists all possible locale values and the respective languages. Typically, only some of these locales will be enabled for a company and available to use with Translation Collections.

Locale	Language
CURRENT	Language currently set by the user
COMPANY_DEFAULT	Default company language
af_ZA	Afrikaans (South Africa)
ar	Arabic
bg_BG	Bulgarian (Bulgaria)
bn_BD	Bengali (Bangladesh)
bs_BA	Bosnian (Bosnia and Herzegovina)
cs_CZ	Czech (Czech Republic)
da_DK	Danish (Denmark)
de_DE	German (Germany)
el_GR	Greek (Greece)
en	English
en_AU	English (Australia)
en_CA	English (Canada)
en_GB	English (United Kingdom)
en_US	English (United States)
es_AR	Spanish (Argentina)
es_ES	Spanish (Spain)
et_EE	Estonian (Estonia)
fa_IR	Farsi (Iran)
fi_FI	Finnish (Finland)
fr_CA	French (Canada)
fr_FR	French (France)
gu_IN	Gujarati (India)
he_IL	Hebrew (Israel)
hi_IN	Hindi (India)
hr_HR	Croatian (Croatia)
hu_HU	Hungarian (Hungary)
hy_AM	Armenian (Armenia)

id_ID	Indonesian (Indonesia)
is_IS	Icelandic (Iceland)
it_IT	Italian (Italy)
ja_JP	Japanese (Japan)
kn_IN	Kannada (India)
ko_KR	Korean (Korea)
lb_LU	Luxembourgish (Luxembourg)
lt_LT	Lithuanian (Lithuania)
lv_LV	Latvian (Latvia)
mr_IN	Marathi (India)
ms_MY	Malay (Malaysia)
nl_NL	Dutch (Netherlands)
no_NO	Norwegian (Norway)
pa_IN	Punjabi (India)
pl_PL	Polish (Poland)
pt_BR	Portuguese (Brazil)
pt_PT	Portuguese (Portugal)
ro_RO	Romanian (Romania)
ru_RU	Russian (Russia)
sh_RS	Serbian (Latin)
sk_SK	Slovakian (Slovakia)
sl_SI	Slovenian (Slovenia)
sq_AL	Albanian (Albania)
sr_RS	Serbian (Serbia)
sv_SE	Swedish (Sweden)
ta_IN	Tamil (India)
te_IN	Telugu (India)
th_TH	Thai (Thailand)
tl_PH	Tagalog (Philippines)
tr_TR	Turkish (Turkey)
uk_UA	Ukrainian (Ukraine)
vi_VN	Vietnamese (Viet Nam)
zh_CN	Chinese (Simplified)

zh_TW

Chinese (Traditional)

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/translation Module Script Samples](#).

```

1 // Add additional code
2 ...
3 var myMsg = message.create({
4   title: translation.get({
5     collection: 'custcollection_my_strings',
6     key: 'MY_TITLE',
7     locale: translation.Locale.COMPANY_DEFAULT
8   })(),
9   message: translation.get({
10    collection: 'custcollection_my_strings',
11    key: 'MY_MESSAGE',
12    locale: translation.Locale.COMPANY_DEFAULT
13  })(),
14   type: message.Type.CONFIRMATION
15 });
16 ...
17 // Add additional code

```

N/ui Modules



Note: The content in this help topic pertains to SuiteScript 2.0.

Modules within the N/ui namespace allow you to build a custom UI using SuiteScript 2.0. Use ['N/ui'] as the first argument of the define function as a shortcut to load the three modules (N/ui/dialog, N/ui/message, and N/ui/serverWidget) on server scripts, or the N/ui/dialog and N/ui/message on client scripts.

- [N/ui/dialog Module](#) — Used to create modal dialog boxes that can be used to present additional options or alerts. This module uses JavaScript promises to manage dialogs asynchronously.
- [N/ui/message Module](#) — Used to display and manage messages at the top of your User Interface.
- [N/ui/serverWidget Module](#) — Contains the UI components used to work with user interfaces in NetSuite. This module is used to create custom pages, forms, lists and widgets that have the NetSuite look-and-feel.

N/ui/dialog Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the dialog module to create a modal dialog that persists until a button on the dialog is pressed.



Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). The NetSuite UI should only be accessed using SuiteScript APIs.

- [N/ui/dialog Module Members](#)

- N/ui/dialog Module Script Samples

N/ui/dialog Module Members

Member Type	Name	Property Type / Method Return Type	Supported Script Types	Description
Method	dialog.alert(options)	Promise	Client scripts	Creates an Alert dialog with an OK button.
	dialog.confirm(options)	Promise	Client scripts	Creates a Confirm dialog with OK and Cancel buttons.
	dialog.create(options)	Promise	Client scripts	Creates a dialog with specified buttons.

N/ui/dialog Module Script Samples

For help with writing scripts in SuiteScript 2.0, see the help topics [SuiteScript 2.0 Hello World](#) and [SuiteScript 2.0 Entry Point Script Creation and Deployment](#).

Sample 1: Create an Alert dialog

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

i Note: To debug client scripts like the following, we recommend that you use Chrome DevTools for Chrome, Firebug debugger for Firefox, or Microsoft Script Debugger for Internet Explorer. For information about these tools, see the documentation provided with each browser. For more information on debugging SuiteScript client scripts, see the help topic [Debugging Client Scripts](#).

The following sample shows how to create an Alert dialog:

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/ui/dialog'],
6   function(dialog) {
7     var options = {
8       title: "I am an Alert",
9       message: "Click OK to continue."
10    };
11    function success(result) {
12      console.log("Success with value " + result);
13    }
14    function failure(reason) {
15      console.log("Failure: " + reason);
16    }
17
18    dialog.alert(options).then(success).catch(failure);
19 });

```

The following screenshot shows the Alert dialog created in this example:



Sample 2: Create a Confirmation dialog

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

i Note: To debug client scripts like the following, we recommend that you use Chrome DevTools for Chrome, Firebug debugger for Firefox, or Microsoft Script Debugger for Internet Explorer. For information about these tools, see the documentation provided with each browser. For more information on debugging SuiteScript client scripts, see the help topic [Debugging Client Scripts](#).

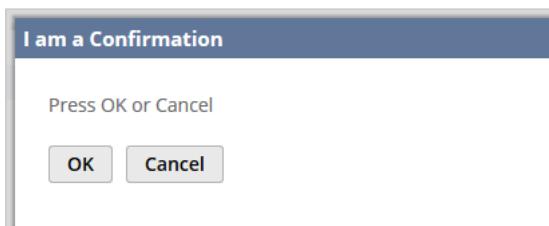
The following sample shows how to create a Confirmation dialog:

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/ui/dialog'],
6   function(dialog) {
7     var options = {
8       title: "I am a Confirmation",
9       message: "Press OK or Cancel"
10    };
11    function success(result) {
12      console.log("Success with value " + result);
13    }
14    function failure(reason) {
15      console.log("Failure: " + reason);
16    }
17
18    dialog.confirm(options).then(success).catch(failure);
19 });

```

The following screenshot shows the Confirmation dialog created in this example:



Sample 3: Create a dialog with custom buttons

Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

Note: To debug client scripts like the following, we recommend that you use Chrome DevTools for Chrome, Firebug debugger for Firefox, or Microsoft Script Debugger for Internet Explorer. For information about these tools, see the documentation provided with each browser. For more information on debugging SuiteScript client scripts, see the help topic [Debugging Client Scripts](#).

The following sample shows how to create a dialog with buttons:

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/ui/dialog'],
6         function(dialog) {
7             var button1 = {
8                 label: 'I am A',
9                 value: 1
10            };
11            var button2 = {
12                label: 'I am B',
13                value: 2
14            };
15            var button3 = {
16                label: 'I am C',
17                value: 3
18            };
19            var options = {
20                title: 'Alphabet Test',
21                message: 'Which One?',
22                buttons: [button1, button2, button3]
23            };
24
25            function success(result) {
26                console.log("Success with value " + result);
27            }
28            function failure(reason) {
29                console.log("Failure: " + reason);
30            }
31            dialog.create(options).then(success).catch(failure);
32        });

```

The following screenshot shows the dialog with buttons created in this example:



Sample 4: Create a dialog with the default button

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

Note: To debug client scripts like the following, we recommend that you use Chrome DevTools for Chrome, Firebug debugger for Firefox, or Microsoft Script Debugger for Internet Explorer. For information about these tools, see the documentation provided with each browser. For more information on debugging SuiteScript client scripts, see the help topic [Debugging Client Scripts](#).

The following sample shows the default behavior when you create a dialog without specifying any buttons, a single button with the label OK.

```

1  /**
2   * @NApiVersion 2.x
3   */
4
5  require(['N/ui/dialog'],
6         function(dialog) {
7             var options = {
8                 title: 'I am a Dialog with the default button',
9                 message: 'Click a button to continue.',
10            };
11
12            function success(result) {
13                console.log("Success with value " + result);
14            }
15            function failure(reason) {
16                console.log("Failure: " + reason);
17            }
18            dialog.create(options).then(success).catch(failure);
19        });

```

The following screenshot shows the dialog with the default button created in this example:



dialog.alert(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates an Alert dialog with an OK button.
Returns	Promise Object. To run a callback function when the OK button is clicked, pass a function to the then portion of the Promise object. When the OK button is clicked, true is passed to the callback. You do not have to utilize the Promise object unless there is an action you want performed after the user clicks the OK button.
Supported Script Types	Client scripts

	For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/ui/dialog Module
Since	2016.1

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	optional	The alert dialog title. This value defaults to an empty string.	2016.1
options.message	string	optional	The content of the alert dialog. This value defaults to an empty string.	2016.1

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/dialog Module Script Samples](#).

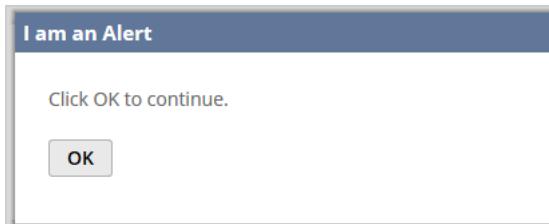
```

1 //Add additional code
2 ...
3     function success(result) { console.log('Success with value: ' + result) }
4     function failure(reason) { console.log('Failure: ' + reason) }

5
6     dialog.alert({
7         title: 'I am an Alert',
8         message: 'Click OK to continue.'
9     }).then(success).catch(failure);
10 ...
11 //Add additional code

```

The following screenshot shows an example of an Alert dialog:



dialog.confirm(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a Confirm dialog with OK and Cancel buttons.
Returns	Promise Object . To run a callback function when the OK button is pressed, pass a function to the then portion of the Promise object. The value of the pressed button, where OK is true and Cancel is false , is passed to the callback.

Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/ui/dialog Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	optional	The confirmation dialog title. This value defaults to an empty string.	2016.1
options.message	string	optional	The content of the confirmation dialog. This value defaults to an empty string.	2016.1

Syntax

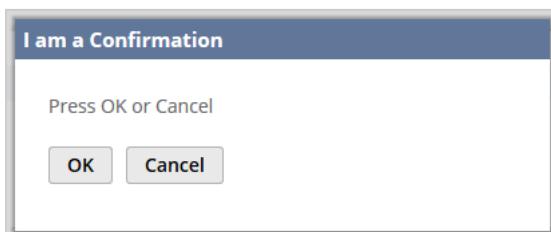
Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/dialog Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var options = {
4   title: "I am a Confirmation",
5   message: "Press OK or Cancel"
6 };
7
8 function success(result) {
9   console.log("Success with value " + result);
10 }
11
12 function failure(reason) {
13   console.log("Failure: " + reason);
14 }
15
16 dialog.confirm(options).then(success).catch(failure);
17
18 ...
19 //Add additional code

```

The following screenshot shows an example of a Confirmation dialog:



dialog.create(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a dialog with specified buttons.
Returns	Promise Object. To run a callback function when a button is pressed, pass a function to the then portion of the Promise object. The value of the button pressed is passed to the callback.
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/ui/dialog Module
Since	2016.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.buttons	string[]	optional	A list of buttons to include in the dialog. Each item in the button list must be a Javascript Object that contains a label and a value property. By default, a single button with the label OK and the value true is used.	2016.1
options.title	string	optional	The dialog title. This value defaults to an empty string.	2016.1
options.message	string	optional	The content of the dialog. This value defaults to an empty string.	2016.1

Errors

Error Code	Error Message	Thrown If
WRONG_PARAMETER_TYPE	Wrong parameter type: {1} is expected as {2}.	The options.buttons parameter is specified and is not an array.
BUTTONS_MUST_INCLUDE_BOTH_A_LABEL_AND_VALUE	Buttons must include both a label and value.	The options.buttons parameter is specified and one or more items do not have a label, a value, or both.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/dialog Module Script Samples](#).

```

1 //Add additional code
2 ...
3     var options = {
4         title: 'I am a Dialog with 3 custom buttons',
5         message: 'Click a button to continue.',
6         buttons: [
7             { label: '1', value: 1 },
8             { label: '2', value: 2 },
9             { label: '3', value: 3 }
10        ];
11
12        function success(result) { console.log('Success with value: ' + result) }
13        function failure(reason) { console.log('Failure: ' + reason) }
14
15        dialog.create(options).then(success).catch(failure);
16        ...
17        //Add additional code

```

The following screenshot shows an example of a Custom dialog with three buttons:



If no buttons are specified, a single value with the label OK is used:

```

1 //Add additional code
2 ...
3     dialog.create({
4         title: 'I am a Dialog with the default button',
5         message: 'Click a button to continue.'
6     });
7 ...
8     //Add additional code

```



N/ui/message Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the message module to display a message at the top of the screen under the menu bar.



Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). The NetSuite UI should only be accessed using SuiteScript APIs.

- [N/ui/message Members](#)
- [Message Object Members](#)
- [N/ui/message Module Script Sample](#)

N/ui/message Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	message.Message	void	Client scripts	Encapsulates the Message object that gets created when calling the create method.
Method	message.create(options)	message.Message	Client scripts	Creates a message that can be displayed or hidden near the top of the page.
Enum	message.Type	enum	Client scripts	Indicates the type of message to display, which specifies the background color of the message and other message indicators.

Message Object Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Message.hide()	void	Client scripts	Hides the message.
	Message.show()	void	Client scripts	Shows the message.

N/ui/message Module Script Sample

For help with writing scripts in SuiteScript 2.0, see the help topics [SuiteScript 2.0 Hello World](#) and [SuiteScript 2.0 Entry Point Script Creation and Deployment](#).



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).



Note: To debug client scripts like the following, we recommend that you use Chrome DevTools for Chrome, Firebug debugger for Firefox, or Microsoft Script Debugger for Internet Explorer. For information about these tools, see the documentation provided with each browser. For more information on debugging SuiteScript client scripts, see the help topic [Debugging Client Scripts](#).

The following sample shows how to create messages for the four available types (confirmation, information, warning, and error).

```
1 | /**
```

```

2  * @NApiVersion 2.x
3  */
4
5  require(['N/ui/message'],
6    function(message) {
7      var myMsg = message.create({
8        title: "My Title",
9        message: "My Message",
10       type: message.Type.CONFIRMATION
11     });
12
13     // will disappear after 5s
14     myMsg.show({
15       duration: 5000
16     });
17
18     var myMsg2 = message.create({
19       title: "My Title 2",
20       message: "My Message 2",
21       type: message.Type.INFORMATION
22     });
23
24     myMsg2.show();
25     setTimeout(myMsg2.hide, 15000); // will disappear after 15s
26
27     var myMsg3 = message.create({
28       title: "My Title 3",
29       message: "My Message 3",
30       type: message.Type.WARNING,
31       duration: 20000
32     );
33
34     myMsg3.show(); // will disappear after 20s
35
36     var myMsg4 = message.create({
37       title: "My Title 4",
38       message: "My Message 4",
39       type: message.Type.ERROR
40     );
41
42     myMsg4.show(); // will stay up until hide is called.
43   });

```

You can see the outcome in the following screenshot:



message.Message

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	The Message object that gets created when calling the create method. For a complete list of this object's methods, see Message Object Members .
Supported Script Types	Client scripts

	For more information, see the help topic SuiteScript 2.0 Client Script Type .
Module	N/ui/message Module
Since	2016.1

Message.hide()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Hides the message.
Returns	void
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/ui/message Module
Since	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/message Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myMsg = message.create({
4   title: "My Title 2",
5   message: "My Message 2",
6   type: message.Type.INFORMATION
7 });
8 myMsg.show();
9 setTimeout(myMsg.hide(), 15000); //hide the message after 15s
10 ...
11 //Add additional code

```

Message.show()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Shows the message.
Returns	void
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/ui/message Module
Since	2016.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.duration	int string	optional	The amount of time, in milliseconds, to show the message. The default is 0, which shows the message until Message.hide() is called. If you use a string, it will be parsed to a number. If you specify a duration for message.create() and message.show(), the value from the message.show() method call takes precedence.	2016.1

Errors

Error Code	Error Message	Thrown If
WRONG_PARAMETER_TYPE	Wrong parameter type: {1} is expected as {2}.	The options.duration is specified with a non-numerical value.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/message Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myMsg = message.create({
4   title: "My Title 2",
5   message: "My Message 2",
6   type: message.Type.INFORMATION
7 });
8 myMsg.show({ duration : 1500 });
9 ...
10 //Add additional code

```

message.create(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a message that can be displayed or hidden near the top of the page.
Returns	message.Message .
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .
Governance	None
Module	N/ui/message Module
Since	2016.1

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.type	message.Type	required	The message type. Set this value using the message.Type enum.	2016.1
options.title	string	optional	The message title. This value defaults to an empty string.	2016.1
options.message	string	optional	The content of the message. This value defaults to an empty string.	2016.1
options.duration	int string	optional	The amount of time, in milliseconds, to show the message. The default is 0, which shows the message until <code>Message.hide()</code> is called. If you use a string, it will be parsed to a number. If you specify a duration for <code>message.create()</code> and <code>message.show()</code> , the value from the <code>message.show()</code> method call takes precedence.	2018.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/message Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myMsg = message.create({
4   title: "My Title",
5   message: "My Message",
6   type: message.Type.CONFIRMATION
7 });
8 ...
9 //Add additional code

```

message.Type

i Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Indicates the type of message to display, which specifies the background color of the message and other message indicators.
	i Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.
Supported Script Types	Client scripts For more information, see the help topic SuiteScript 2.0 Client Script Type .

Module	N/ui/message Module
Since	2016.1

Values

Value	Color
CONFIRMATION	A green background with a checkmark icon.
INFORMATION	A blue background with an Information icon.
WARNING	A yellow background with a Warning icon.
ERROR	A red background with an X icon.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/message Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var myMsg = message.create({
4   title: "My Title",
5   message: "My Message",
6   type: message.Type.CONFIRMATION
7 });
8 myMsg.show();
9 ...
10 //Add additional code

```

N/ui/serverWidget Module

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Load the serverWidget module when you want to work with the user interface within NetSuite. You can use Suitelets to build custom pages and wizards that have a NetSuite look-and-feel. You can also create various components of the NetSuite UI (for example, forms, fields, sublists, tabs).

 **Important:** SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). The NetSuite UI should only be accessed using SuiteScript APIs .

 **Important:** When you add a UI object to an existing NetSuite page, to minimize the occurrence of field/object name conflicts, the internal ID that references the object must be prefixed with custpage.

- [N/ui/serverWidget Module Members](#)
- [Assistant Object Members](#)
- [AssistantStep Object Members](#)
- [Button Object Members](#)
- [Field Object Members](#)
- [FieldGroup Object Members](#)

- Form Object Members
- List Object Members
- ListColumn Object Members
- Sublist Object Members
- Tab Object Members
- N/ui/serverWidget Module Script Samples

N/ui/serverWidget Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	serverWidget.Assistant	Object	Suitelets and beforeLoad user events	A scriptable, multi-step NetSuite assistant.
	serverWidget.AssistantStep	Object	Suitelets and beforeLoad user events	A step within a custom NetSuite assistant.
	serverWidget.Button	Object	Suitelets and beforeLoad user events	A button that appears in a UI object.
	serverWidget.Field	Object	Suitelets and beforeLoad user events	A NetSuite field.
	serverWidget.FieldGroup	Object	Suitelets and beforeLoad user events	A field group.
	serverWidget.Form	Object	Suitelets and beforeLoad user events	A NetSuite form.
	serverWidget.List	Object	Suitelets and beforeLoad user events	A list.
	serverWidget.ListColumn	Object	Suitelets and beforeLoad user events	A list column.
	serverWidget.Sublist	Object	Suitelets and beforeLoad user events	A NetSuite sublist.
	serverWidget.Tab	Object	Suitelets and beforeLoad user events	A NetSuite tab and subtabs.
Method	serverWidget.createAssistant(options)	serverWidget.Assistant	Suitelets and beforeLoad user events	Creates and returns a new assistant object.
	serverWidget.createForm(options)	serverWidget.Form	Suitelets and beforeLoad user events	Creates and returns a new form object.
	serverWidget.createList(options)	serverWidget.List	Suitelets and beforeLoad user events	Creates a List object (specifying the title, and whether to hide the navigation bar).

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Enum	serverWidget.AssistantSubmitAction	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for submit actions performed by the user.
	serverWidget.FieldBreakType	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for supported field break types. This enum is used to set the value of the Field.updateBreakType(options) property.
	serverWidget.FieldDisplayType	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for supported field display types. This enum is used to set the value of the Field.updateDisplayType(options) property.
	serverWidget.FieldLayoutType	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for the supported types of field layouts. This enum is used to set the value of the Field.updateLayoutType(options) property.
	serverWidgetFieldType	string (read-only)	Suitelets and beforeLoad user events	Holds the values for supported field types. This enum is used to set the value of the Field.type property.
	serverWidgetFormPageLinkType	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for supported page link types on a form. This enum is used to set the value of the type parameter for Form.addPageLink(options) .
	serverWidgetLayoutJustification	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for supported justification layouts. This enum is used to set the value of the align parameter when List.addColumn(options) is called.
	serverWidgetListStyle	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for supported list styles. This enum is used to set the value of the List.style property.
	serverWidgetSublistDisplayType	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for supported sublist display types. This enum is used to set the value of the Sublist.displayType property.
	serverWidgetSublistType	string (read-only)	Suitelets and beforeLoad user events	Holds the string values for valid sublist types. This enum is used to define the type parameter when Form.addSublist(options) is called.

Assistant Object Members

The following members are called on the [serverWidget.Assistant](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Assistant.addField(options)	serverWidget.Field	Suitelets and beforeLoad user events	Adds a field to an assistant.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Assistant.addFieldGroup(options)	serverWidget.FieldGroup	Suitelets and beforeLoad user events	Adds a field group to an assistant.
	Assistant.addStep(options)	serverWidget.AssistantStep	Suitelets and beforeLoad user events	Adds a step to an assistant.
	Assistant.addSublist(options)	serverWidget.Sublist	Suitelets and beforeLoad user events	Adds a sublist to an assistant.
	Assistant.getField(options)	serverWidget.Field	Suitelets and beforeLoad user events	Gets a field object.
	Assistant.getFieldGroup(options)	serverWidget.FieldGroup	Suitelets and beforeLoad user events	Gets a field group object.
	Assistant.getFieldGroupIds()	string[]	Suitelets and beforeLoad user events	Gets all the field group IDs in an assistant.
	Assistant.getFieldIds()	string[]	Suitelets and beforeLoad user events	Gets all the field IDs in an assistant.
	Assistant.getFieldIdsByFieldGroup(fieldGroup)	string[]	Suitelets and beforeLoad user events	Gets all field IDs in the assistant field group.
	Assistant.getLastAction()	string	Suitelets and beforeLoad user events	Gets the last action submitted by the user.
	Assistant.getLastStep()	serverWidget.AssistantStep	Suitelets and beforeLoad user events	Gets the step that the last submitted action came from.
	Assistant.getNextStep()	serverWidget.AssistantStep	Suitelets and beforeLoad user events	Gets the next step prompted by the assistant.
	Assistant.getStep(options)	serverWidget.AssistantStep	Suitelets and beforeLoad user events	Returns a step in an assistant.
	Assistant.getStepCount()	number	Suitelets and beforeLoad user events	Gets the total count of steps in the assistant.
	Assistant.getSteps()	serverWidget.AssistantStep[]	Suitelets and beforeLoad user events	Gets all the steps in the assistant.
	Assistant.getSublist(options)	serverWidget.Sublist	Suitelets and beforeLoad user events	Get a Sublist object from its ID.
	Assistant.getSublistIds()	string[]	Suitelets and beforeLoad user events	Gets all the sublist IDs in an assistant.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Assistant.hasErrorHtml()	boolean	Suitelets and beforeLoad user events	Indicates whether the assistant has an error message to display.
	Assistant.isFinished()	boolean	Suitelets and beforeLoad user events	Indicates the status of the assistant. If set to true, the assistant is finished.
	Assistant.sendRedirect(options)	void	Suitelets and beforeLoad user events	Manages redirects in an assistant.
	Assistant.setSplash(options)	void	Suitelets and beforeLoad user events	Define a splash message.
	Assistant.updateDefaultValues(values)	void	Suitelets and beforeLoad user events	Sets the default values of an array of fields that are specific to the assistant.
Property	Assistant.clientScriptFileDialog	number	Suitelets and beforeLoad user events	The File Cabinet ID of client script file to be used in this assistant.
	Assistant.clientScriptModulePath	string	Suitelets and beforeLoad user events	The relative path to the client script file to be used in this assistant.
	Assistant.currentStep	serverWidget.AssistantStep (read-only)	Suitelets and beforeLoad user events	The current step.
	Assistant.errorHtml	string	Suitelets and beforeLoad user events	The error message text.
	Assistant.finishedHtml	string	Suitelets and beforeLoad user events	The text displayed after an assistant is finished.
	Assistant.hideAddToShortcutsLink	boolean	Suitelets and beforeLoad user events	Whether the Add to Shortcuts Link is displayed in the UI.
	Assistant.hideStepNumber	boolean	Suitelets and beforeLoad user events	Whether the current and total step numbers are displayed in the UI.
	Assistant.isNotOrdered	boolean	Suitelets and beforeLoad user events	Whether assistant steps are ordered or unordered.
	Assistant.title	string	Suitelets and beforeLoad user events	The title of the assistant.

AssistantStep Object Members

The following members are called on the [serverWidget.AssistantStep](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	AssistantStep.getFieldIds()	string[]	Suitelets and beforeLoad user events	Gets all the field IDs in an assistant step.
	AssistantStep.getLineCount(options)	number	Suitelets and beforeLoad user events	Gets the number of lines previously entered by a user in a step.
	AssistantStep.getLineCount(options)	string[]	Suitelets and beforeLoad user events	Gets all the field IDs in a list.
	AssistantStep.getSublistValue(options)	string	Suitelets and beforeLoad user events	Gets the current value of a sublist field (line item) in a step.
	AssistantStep.getSubmittedSublistIds()	string[]	Suitelets and beforeLoad user events	Gets the IDs for all the sublist fields (line items) in a step.
	AssistantStep.getValue(options)	string string[]	Suitelets and beforeLoad user events	Gets the current value of a field.
Property	AssistantStep.helpText	string	Suitelets and beforeLoad user events	The help text for a step.
	AssistantStep.id	string (read-only)	Suitelets and beforeLoad user events	The internal ID of the step.
	AssistantStep.label	string	Suitelets and beforeLoad user events	The label for a step.
	AssistantStep.stepNumber	number	Suitelets and beforeLoad user events	Indicates where this step appears sequentially in an assistant.

Button Object Members

The following members are called on the [serverWidget.Button](#) object.

Member Type	Name	Property Type	Supported Script Types	Description
Property	Button.isDisabled	boolean	Suitelets and beforeLoad user events	Whether a button is grayed-out and disabled.
	Button.isHidden	boolean	Suitelets and beforeLoad user events	Whether the button is hidden in the UI.
	Button.label	string	Suitelets and beforeLoad user events	The label for the button.

Field Object Members

The following members are called on the [serverWidget.Field](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Field.addSelectOption(options)	void	Suitelets and beforeLoad user events	Adds a select option to a dropdown list for a selectable field.
	Field.getSelectOptions(options)	Object[]	Suitelets and beforeLoad user events	Returns the internal ID and label of the options for a select field as name/value pairs.
	Field.setHelpText(options)	serverWidget.Field	Suitelets and beforeLoad user events	Sets the help text that appears in the field help popup.
	Field.updateBreakType(options)	serverWidget.Field	Suitelets and beforeLoad user events	Updates the break type used to add a break in flow layout for the field.
	Field.updateDisplaySize(options)	serverWidget.Field	Suitelets and beforeLoad user events	Updates the height and width for the field.
	Field.updateDisplayType(options)	serverWidget.Field	Suitelets and beforeLoad user events	Updates the type of display for the field.
	Field.updateLayoutType(options)	serverWidget.Field	Suitelets and beforeLoad user events	Updates the layout type for the field.
Property	Field.alias	string	Suitelets and beforeLoad user events	The alias used to set the field value.
	Field.defaultValue	string	Suitelets and beforeLoad user events	The default value for the field.
	Field.helpText	string (read-only)	Suitelets and beforeLoad user events	The help text for the field.
	Field.id	string (read-only)	Suitelets and beforeLoad user events	The internal ID for the field.
	Field.isMandatory	boolean	Suitelets and beforeLoad user events	Whether the field is mandatory.
	Field.label	string	Suitelets and beforeLoad user events	The label for the field.
	Field.linkText	string	Suitelets and beforeLoad user events	The text displayed for a link in place of the URL.
	Field.maxLength	number	Suitelets and beforeLoad user events	The maximum length, in characters, for the field.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Field.padding	number	Suitelets and beforeLoad user events	The number of empty vertical character spaces above the field.
	Field.richTextHeight	number	Suitelets and beforeLoad user events	The height of a rich text field, in pixels.
	Field.richTextWidth	number	Suitelets and beforeLoad user events	The width of a rich text field, in pixels.
	Field.type	string (read-only)	Suitelets and beforeLoad user events	The type of field.

FieldGroup Object Members

The following members are called on the [serverWidget.FieldGroup](#) object.

Member Type	Name	Property Type	Supported Script Types	Description
Property	FieldGroup.isBorderHidden	boolean	Suitelets and beforeLoad user events	Whether a border appears around the field group.
	FieldGroup.isCollapsible	boolean	Suitelets and beforeLoad user events	Whether the field group is collapsible.
	FieldGroup.isCollapsed	boolean	Suitelets and beforeLoad user events	Whether the field group is initially collapsed or expanded in the default view.
	FieldGroup.isSingleColumn	boolean	Suitelets and beforeLoad user events	Whether the field group is displayed in a single column.
	FieldGroup.label	string	Suitelets and beforeLoad user events	The label for the field group.

Form Object Members

The following members are called on the [serverWidget.Form](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Form.addButton(options)	serverWidget.Button	Suitelets and beforeLoad user events	Adds a button to the form.
	Form.addCredentialField(options)	serverWidget.Field	Suitelets and beforeLoad user events	Adds a field that stores credentials in NetSuite for invoking services provided by third parties.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Form.addField(options)	serverWidget.Field	Suitelets and beforeLoad user events	Adds a field to the form.
	Form.addFieldGroup(options)	serverWidget.FieldGroup	Suitelets and beforeLoad user events	Adds a group of fields to the form.
	Form.addPageInitMessage(options)	void	Suitelets and beforeLoad user events	Shows a message on a form in view mode. You can use this method to show a message on a form based on its user event script context.
	Form.addPageLink(options)	void	Suitelets and beforeLoad user events	Adds a link to a form.
	Form.addResetButton(options)	serverWidget.Button	Suitelets and beforeLoad user events	Adds a reset button to a form that clears user input.
	Form.addSecretKeyField(options)	serverWidget.Field	Suitelets and beforeLoad user events	Add a secret key field to the form.
	Form.addSublist(options)	serverWidget.Sublist	Suitelets and beforeLoad user events	Adds a sublist to the form.
	Form.addButton(options)	serverWidget.Button	Suitelets and beforeLoad user events	Adds a submit button to a form that saves user inputs.
	Form.addSubtab(options)	serverWidget.Tab	Suitelets and beforeLoad user events	Adds a subtab to a form.
	Form.addTab(options)	serverWidget.Tab	Suitelets and beforeLoad user events	Adds a tab to a form.
	Form.getButton(options)	serverWidget.Button	Suitelets and beforeLoad user events	Returns a button by internal ID.
	Form.getField(options)	serverWidget.Field	Suitelets and beforeLoad user events	Returns a field by internal ID.
	Form.getSublist(options)	serverWidget.Sublist	Suitelets and beforeLoad user events	Returns a sublist by internal ID.
	Form.getSubtab(options)	serverWidget.Tab	Suitelets and beforeLoad user events	Returns a subtab by internal ID.
	Form.getTab(options)	serverWidget.Tab	Suitelets and beforeLoad user events	Returns a tab object from its internal ID.
	Form.getTabs()	serverWidget.Tab[]	Suitelets and beforeLoad user events	Returns an array of all the tabs in a form.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Form.insertField(options)	void	Suitelets and beforeLoad user events	Inserts a field before another field within a form.
	Form.insertSublist(options)	void	Suitelets and beforeLoad user events	Inserts a sublist before another sublist on a form.
	Form.insertSubtab(options)	void	Suitelets and beforeLoad user events	Inserts a subtab before another subtab on a form.
	Form.insertTab(options)	void	Suitelets and beforeLoad user events	Inserts a tab before another tab on a form.
	Form.removeButton(options)	void	Suitelets and beforeLoad user events	Removes a button from a form.
	Form.updateDefaultValues(options)	void	Suitelets and beforeLoad user events	Sets the default values of many fields on a form.
Property	Form.clientScriptFileDialog	number	Suitelets and beforeLoad user events	The File Cabinet ID of client script file to be used in this form.
	Form.clientScriptModulePath	string	Suitelets and beforeLoad user events	The relative path to the client script file to be used in this form.
	Form.title	string	Suitelets and beforeLoad user events	The title used for the form.

List Object Members

The following members are called on the [serverWidget.List](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	List.addButton(options)	serverWidget.Button	Suitelets and beforeLoad user events	Adds a button to a list.
	List.addColumn(options)	serverWidget.ListColumn	Suitelets and beforeLoad user events	Adds a column to a list.
	List.addEditColumn(options)	serverWidget.ListColumn	Suitelets and beforeLoad user events	Adds a column containing Edit or Edit/View links to a Suitelet or Portlet list.
	List.addPageLink(options)	serverWidget.List	Suitelets and beforeLoad user events	Adds a link to a list.
	List.addRow(options)	serverWidget.List	Suitelets and beforeLoad user events	Adds a single row to a list.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	List.addRows(options)	serverWidget.List	Suitelets and beforeLoad user events	Adds multiple rows to a list.
Property	List.clientScriptFileDialog	number	Suitelets and beforeLoad user events	The File Cabinet ID of client script file to be used in this list.
	List.clientScriptModulePath	string	Suitelets and beforeLoad user events	The relative path to the client script file to be used in this list.
	List.style	string	Suitelets and beforeLoad user events	The display style for this list.
	List.title	string	Suitelets and beforeLoad user events	The List title.

ListColumn Object Members

The following members are called on the [serverWidget.ListColumn](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	ListColumn.addParamToURL(options)	serverWidget.ListColumn	Suitelets and beforeLoad user events	Adds a URL parameter (optionally defined per row) to the list column's URL.
	ListColumn.setURL(options)	serverWidget.ListColumn	Suitelets and beforeLoad user events	Sets the base URL for the list column.
Property	ListColumn.label	string	Suitelets and beforeLoad user events	The label of this list column.

Sublist Object Members

The following members are called on the [serverWidget.Sublist](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Sublist.addButton(options)	serverWidget.Button	Suitelets and beforeLoad user events	Adds a button to a sublist.
	Sublist.addField(options)	serverWidget.Field	Suitelets and beforeLoad user events	Add a field to a sublist.
	Sublist.addMarkAllButtons()	serverWidget.Button[]	Suitelets and beforeLoad user events	Adds a Mark All or Unmark All button.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Sublist.addRefreshButton()	serverWidget.Button	Suitelets and beforeLoad user events	Adds a Reset button.
	Sublist.getField(options)	serverWidget.Field	Suitelets and beforeLoad user events	Returns a Field object on a specified sublist.
	Sublist.getSublistValue(options)	string	Suitelets and beforeLoad user events	Gets a field value on a sublist.
	Sublist.setSublistValue(options)	void	Suitelets and beforeLoad user events	Sets the value of a sublist field.
	Sublist.updateTotallingFieldId(options)	serverWidget.Sublist	Suitelets and beforeLoad user events	Updates the ID of a field designated as a totalling column, which is used to calculate and display a running total for the sublist.
	Sublist.updateUniqueFieldId(options)	serverWidget.Sublist	Suitelets and beforeLoad user events	Updates a field ID that is to have unique values across the rows in the sublist.
Property	Sublist.displayType	string	Suitelets and beforeLoad user events	The display style for a sublist.
	Sublist.helpText	string	Suitelets and beforeLoad user events	The inline help text for a sublist.
	Sublist.label	string	Suitelets and beforeLoad user events	The label for a sublist.
	Sublist.lineCount	number (read-only)	Suitelets and beforeLoad user events	The number of line items in a sublist.

Tab Object Members

The following members are called on the [serverWidget.Tab](#) object.

Member Type	Name	Value Type	Supported Script Types	Description
Property	Tab.helpText	string	Suitelets and beforeLoad user events	The inline help text for a tab or subtab.
	Tab.label	string	Suitelets and beforeLoad user events	The label for a tab or subtab.

N/ui/serverWidget Module Script Samples

For help with writing scripts in SuiteScript 2.0, see the help topics [SuiteScript 2.0 Hello World](#) and [SuiteScript 2.0 Entry Point Script Creation and Deployment](#).

See the help topic [Complete Custom List Page Sample Script](#) for a sample of a List page, and [Sample Custom Assistant Script](#) for a sample of an Assistant.

Sample 1: Create a Suitelet that generates a sample form

i Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample creates a Suitelet that generates a sample form with a submit button, fields, and an inline editor sublist.

```

1  /**
2   * @NApiVersion 2.x
3   * @NScriptType Suitelet
4   */
5 define(['N/ui/serverWidget'], function(serverWidget) {
6     function onRequest(context) {
7       if (context.request.method === 'GET') {
8         var form = serverWidget.createForm({
9           title: 'Simple Form'
10        });
11
12        var field = form.addField({
13          id: 'custpage_text',
14          type: serverWidget.FieldType.TEXT,
15          label: 'Text'
16        });
17        field.layoutType = serverWidget.FieldLayoutType.NORMAL;
18        field.updateBreakType({
19          breakType: serverWidget.FieldBreakType.STARTCOL
20        });
21
22        form.addField({
23          id: 'custpage_date',
24          type: serverWidget.FieldType.DATE,
25          label: 'Date'
26        });
27        form.addField({
28          id: 'custpage_currencyfield',
29          type: serverWidget.FieldType.CURRENCY,
30          label: 'Currency'
31        });
32
33        var select = form.addField({
34          id: 'custpage_selectfield',
35          type: serverWidget.FieldType.SELECT,
36          label: 'Select'
37        });
38        select.addSelectOption({
39          value: 'a',
40          text: 'Albert'
41        });
42        select.addSelectOption({
43          value: 'b',
44          text: 'Baron'
45        });
46
47        var sublist = form.addSublist({
48          id: 'sublist',
49          type: serverWidget.SublistType.INLINEEDITOR,
50          label: 'Inline Editor Sublist'
51        });
52        sublist.addField({
53          id: 'sublist1',
54          type: serverWidget.FieldType.DATE,
55          label: 'Date'
56        });
57        sublist.addField({
58          id: 'sublist2',
59          type: serverWidget.FieldType.TEXT,
60          label: 'Text'
61      });
62    }
63  });

```

```

61    });
62
63    form.addSubmitButton({
64      label: 'Submit Button'
65    });
66
67    context.response.writePage(form);
68  } else {
69    var delimiter = /\u0001/;
70    var textField = context.request.parameters.textfield;
71    var dateField = context.request.parameters.datefield;
72    var currencyField = context.request.parameters.currencyfield;
73    var selectField = context.request.parameters.selectfield;
74    var sublistData = context.request.parameters.sublistdata.split(delimiter);
75    var sublistField1 = sublistData[0];
76    var sublistField2 = sublistData[1];
77
78    context.response.write('You have entered: ' + textField + ' ' + dateField + ' '
79      + currencyField + ' ' + selectField + ' ' + sublistField1 + ' ' + sublistField2);
80  }
81}
82
83 return {
84   onRequest: onRequest
85 };
86});

```

Sample 2: Create a Suitelet that generates a customer survey form

i Note: This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

The following sample creates Suitelet that generates a customer survey form with inline HTML fields, radio fields, a submit button, and a reset button.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4 */
5 define(['N/ui/serverWidget'], function(serverWidget) {
6   function onRequest(context) {
7     var form = serverWidget.createForm({
8       title: 'Thank you for your interest in Wolfe Electronics',
9       hideNavBar: true
10    });
11
12    var htmlHeader = form.addField({
13      id: 'custpage_header',
14      type: serverWidget.FieldType.INLINEHTML,
15      label: ''
16    }).updateLayoutType({
17      layoutType: serverWidget.FieldLayoutType.OUTSIDEABOVE
18    }).updateBreakType({
19      breakType: serverWidget.FieldBreakType.STARTROW
20    }).defaultValue = '<p style=\\"font-size:20px\\>We pride ourselves on providing the best' +
21      ' services and customer satisfaction. Please take a moment to fill out our survey.</p><br><br>';
22
23    var htmlInstruct = form.addField({
24      id: 'custpage_p1',
25      type: serverWidget.FieldType.INLINEHTML,
26      label: ''
27    }).updateLayoutType({
28      layoutType: serverWidget.FieldLayoutType.OUTSIDEABOVE
29    }).updateBreakType({
30      breakType: serverWidget.FieldBreakType.STARTROW
31    }).defaultValue = '<p style=\\"font-size:14px\\>When answering questions on a scale of 1 to 10,' +
32      ' 1 = Greatly Unsatisfied and 10 = Greatly Satisfied.</p><br><br>';

```

```

33
34     var productRating = form.addField({
35         id: 'custpage_lblproductrating',
36         type: serverWidget.FieldType.INLINEHTML,
37         label: ''
38     }).updateLayoutType({
39         layoutType: serverWidget.FieldLayoutType.NORMAL
40     }).updateBreakType({
41         breakType: serverWidget.FieldBreakType.STARTROW
42     }).defaultValue = '<p style="font-size:14px">How would you rate your satisfaction with our products?</p>';
43
44     form.addField({
45         id: 'custpage_rdoprodproductrating',
46         type: serverWidget.FieldType.RADIO,
47         label: '1',
48         source: 'p1'
49     }).updateLayoutType({
50         layoutType: serverWidget.FieldLayoutType.STARTROW
51     });
52     form.addField({
53         id: 'custpage_rdoprodproductrating',
54         type: serverWidget.FieldType.RADIO,
55         label: '2',
56         source: 'p2'
57     }).updateLayoutType({
58         layoutType: serverWidget.FieldLayoutType.MIDROW
59     });
60     form.addField({
61         id: 'custpage_rdoprodproductrating',
62         type: serverWidget.FieldType.RADIO,
63         label: '3',
64         source: 'p3'
65     }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
66     form.addField({
67         id: 'custpage_rdoprodproductrating',
68         type: serverWidget.FieldType.RADIO,
69         label: '4',
70         source: 'p4'
71     }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
72     form.addField({
73         id: 'custpage_rdoprodproductrating',
74         type: serverWidget.FieldType.RADIO,
75         label: '5',
76         source: 'p5'
77     }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
78     form.addField({
79         id: 'custpage_rdoprodproductrating',
80         type: serverWidget.FieldType.RADIO,
81         label: '6',
82         source: 'p6'
83     }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
84     form.addField({
85         id: 'custpage_rdoprodproductrating',
86         type: serverWidget.FieldType.RADIO,
87         label: '7',
88         source: 'p7'
89     }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
90     form.addField({
91         id: 'custpage_rdoprodproductrating',
92         type: serverWidget.FieldType.RADIO,
93         label: '8',
94         source: 'p8'
95     }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
96     form.addField({
97         id: 'custpage_rdoprodproductrating',
98         type: serverWidget.FieldType.RADIO,
99         label: '9',
100        source: 'p9'
101    }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
102    form.addField({
103        id: 'custpage_rdoprodproductrating',
104        type: serverWidget.FieldType.RADIO,
105        label: '10',
106    })

```

```

106     source: 'p10'
107   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.ENDROW});
108
109   var serviceRating = form.addField({
110     id: 'custpage_lblservicerating',
111     type: serverWidget.FieldType.INLINEHTML,
112     label: ''
113   }).updateLayoutType({
114     layoutType: serverWidget.FieldLayoutType.NORMAL
115   }).updateBreakType({
116     breakType: serverWidget.FieldBreakType.STARTROW
117   }).defaultValue = '<p style="font-size:14px">How would you rate your satisfaction with our services?</p>';
118
119   form.addField({
120     id: 'custpage_rdoservicerating',
121     type: serverWidget.FieldType.RADIO,
122     label: '1',
123     source: 'p1'
124   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.STARTROW});
125   form.addField({
126     id: 'custpage_rdoservicerating',
127     type: serverWidget.FieldType.RADIO,
128     label: '2',
129     source: 'p2'
130   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
131   form.addField({
132     id: 'custpage_rdoservicerating',
133     type: serverWidget.FieldType.RADIO,
134     label: '3',
135     source: 'p3'
136   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
137   form.addField({
138     id: 'custpage_rdoservicerating',
139     type: serverWidget.FieldType.RADIO,
140     label: '4',
141     source: 'p4'
142   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
143   form.addField({
144     id: 'custpage_rdoservicerating',
145     type: serverWidget.FieldType.RADIO,
146     label: '5',
147     source: 'p5'
148   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
149   form.addField({
150     id: 'custpage_rdoservicerating',
151     type: serverWidget.FieldType.RADIO,
152     label: '6',
153     source: 'p6'
154   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
155   form.addField({
156     id: 'custpage_rdoservicerating',
157     type: serverWidget.FieldType.RADIO,
158     label: '7',
159     source: 'p7'
160   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
161   form.addField({
162     id: 'custpage_rdoservicerating',
163     type: serverWidget.FieldType.RADIO,
164     label: '8',
165     source: 'p8'
166   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
167   form.addField({
168     id: 'custpage_rdoservicerating',
169     type: serverWidget.FieldType.RADIO,
170     label: '9',
171     source: 'p9'
172   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.MIDROW});
173   form.addField({
174     id: 'custpage_rdoservicerating',
175     type: serverWidget.FieldType.RADIO,
176     label: '10',
177     source: 'p10'
178   }).updateLayoutType({layoutType: serverWidget.FieldLayoutType.ENDROW});

```

```

179     form.addSubmitButton({
180         label: 'Submit'
181     });
182     form.addResetButton({
183         label: 'Reset'
184     });
185
186     context.response.writePage(form);
187 }
188
189 return {
190     onRequest: onRequest
191 };
192 });
193 });

```

serverWidget.Assistant



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>A scriptable, multi-step NetSuite assistant. An assistant contains a series of step that a user must complete to accomplish a larger goal. An assistant can be sequential, or non-sequential and include optional steps. Each page of the assistant is defined by a step.</p> <p>All data and states for an assistant are tracked automatically throughout the user's session until completion of the assistant.</p> <p>You can create a new assistant with the serverWidget.createAssistant(options) method.</p> <p>After you create an Assistant object, you can:</p> <ul style="list-style-type: none"> ▪ Build and run an assistant in your NetSuite account. ▪ Add a variety of scriptable elements to the assistant including fields, steps, buttons, tabs, and sublists. <p>For a complete list of this object's methods and properties, see Assistant Object Members.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4     title : 'Simple Assistant'
5 });
6 ...
7 //Add additional code

```

Assistant.addField(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a field to an assistant. Use fields to record or display information specific to your needs.
Returns	<code>serverWidget.Field</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.id</code>	string	required	The internal ID for this field.	2015.2
<code>options.label</code>	string	required	The label for this field.	2015.2
<code>options.type</code>	string	required	<p>The field type. Use the serverWidget.FieldType enum to set this value.</p> <p>Note: If you have set the type parameter to SELECT, and you want to add custom options to the select field, you must set source to NULL. Then, when a value is specified, the value will populate the options from the source.</p> <p>Important: Long text fields created with SuiteScript have a character limit of 100,000. Long text fields created with Suitebuilder have a character limit of 1,000,000.</p>	2015.2
<code>options.source</code>	string	optional	<p>The internalId or scriptId of the source list for this field. Use this parameter if you are adding a select (List/Record) or multi-select type of field.</p> <p>For radio fields only, the <code>source</code> parameter is not an optional parameter, it must contain the radio button's unique internal ID. The <code>id</code> parameter contains the ID that identifies all the radio buttons of the same group.</p>	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p>Note: If you want to add custom options on a select field, you must set the source parameter to NULL , and then add the custom options using <code>Field.addSelectOption(options)</code>.</p> <p>Important: After you create a select or multi-select field that is sourced from a record or list, you cannot add additional values with <code>Field.addSelectOption(options)</code>. The select values are determined by the source record or list.</p>	
options.container	string	optional	The internal ID of the field group to place this field in.	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addField({
7   id : 'idname',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Sample label'
10 });
11 ...
12 //Add additional code

```

Assistant.addFieldGroup(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a field group to the assistant. A field group is a collection of fields that can be displayed in a one or two column format. Assign a field to a field group in order to label, hide or collapse a group of fields. By default, the field group is collapsible and appears expanded on the assistant page. To change this behavior, set the <code>FieldGroup.isCollapsed</code> and <code>FieldGroup.isCollapsible</code> properties.
Returns	<code>serverWidget.FieldGroup</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID for the field group.	2015.2
options.label	string	required	The label for the field group.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addFieldGroup({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 ...
11 //Add additional code

```

Assistant.addStep(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a step to an assistant. Steps define each page of the assistant. Use Assistant.isNotOrdered to control if the steps must be completed sequentially or in no specific order. If you want to create help text for the step, you can use AssistantStep.helpText on the object returned.
Returns	serverWidget.AssistantStep object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID for this step (for example, 'entercontacts').	2015.2

Parameter	Type	Required / Optional	Description	Since
options.label	string	required	The label for this step (for example, 'Enter Contacts'). By default, the step appears vertically in the left panel of the assistant.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 ...
11 //Add additional code

```

Assistant.addSublist(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a sublist to an assistant.
	<p>Note: Only inline editor sublists are added. Other sublist types are not supported.</p>
Returns	serverWidget.Sublist object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID for the sublist.	2015.2
options.label	string	required	The label for the sublist.	2015.2
options.type	string	required	The type of sublist to add. Currently, only the sublist type of INLINEEDITOR can be added. For	2016.1

Parameter	Type	Required / Optional	Description	Since
			more information about this type of sublist, see serverWidget.SublistType .	

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addSublist({
7   id : 'idname',
8   label : 'Sample label',
9   type : serverWidget.SublistType.INLINEEDITOR
10 });
11 ...
12 //Add additional code

```

Assistant.getField(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a field object on an assistant page.
Returns	serverWidget.Field
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of the field.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```
1 //Add additional code
```

```

2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addField({
7   id : 'idname',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Sample label'
10});
11 var field = assistant.getField({
12   id: 'idname'
13 });
14 ...
15 //Add additional code

```

Assistant.getFieldGroup(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a field group on an assistant page.
Returns	serverWidget.FieldGroup object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of the field group.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addFieldGroup({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var fieldgroup = assistant.getFieldGroup({
11   id: 'idname'
12 });

```

```

13 | ...
14 | //Add additional code

```

Assistant.getFieldGroupIds()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves all the internal IDs for field groups in an assistant.
Returns	string[]
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addFieldGroup({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var fieldgroupid = assistant.getFieldGroupIds();
11 ...
12 //Add additional code

```

Assistant.getFieldIds()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets all the internal IDs for fields in an assistant.
Returns	string[]
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addField({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var fieldid = assistant.getFieldIds();
11 ...
12 //Add additional code

```

Assistant.getFieldIdsByFieldGroup(fieldGroup)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets all field IDs in the assistant field group.
Returns	string[]
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
fieldGroup	string	required	The internal ID of the field group.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addFieldGroup({
7   id : 'fieldgroupid',
8   label : 'Sample label'
9 });
10 assistant.addField({

```

```

11     id : 'idname',
12     label : 'Sample label'
13   });
14   var fieldid = assistant.getFieldIdsByFieldGroup({
15     fieldGroup : 'fieldgroupid'
16   });
17   ...
18 //Add additional code

```

Assistant.getLastAction()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the last action taken by the user. To identify the step that the last action came from, use Assistant.getLastStep() .
Returns	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 ...
7 if (assistant.getLastAction() === serverWidget.AssistantSubmitAction.CANCEL) {
8   ...
9 }
10 ...
11 //Add additional code

```

Assistant.getLastStep()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the step that the last submitted action came from.
Returns	A serverWidget.AssistantStep object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))

Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 ...
7 var lastStep = assistant.getLastStep();
8 ...
9 //Add additional code

```

Assistant.getNextStep()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the next step corresponding to the user's last submitted action in the assistant. If you need information about the last step, use Assistant.getLastStep() before you use this method.
Returns	serverWidget.AssistantStep object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 ...
7 var nextStep = assistant.getNextStep();
8 ...
9 //Add additional code

```

Assistant.getStep(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a step in an assistant.
Returns	<code>serverWidget.AssistantStep</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.id</code>	string	required	The internal ID of the step.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var firststep = assistant.getStep({
11   id: 'idname'
12 });
13 ...
14 //Add additional code

```

Assistant.getStepCount()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the total number of steps in an assistant.
Returns	number
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)

Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var numSteps = assistant.getStepCount();
11 ...
12 //Add additional code

```

Assistant.getSteps()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets all the steps in an assistant.
Returns	serverWidget.AssistantStep[] object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var steps = assistant.getSteps();
11 ...

```

```
12 //Add additional code
```

Assistant.getSublist(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a sublist in an assistant.
Returns	serverWidget.Sublist object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of the sublist.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addSublist({
7   id : 'idname',
8   label : 'Sample label',
9   type : serverWidget.SublistType.LIST
10});
11 var sublist = assistant.getSublist({
12   id : 'idname'
13 });
14 ...
15 //Add additional code

```

Assistant.getSublistIds()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the IDs for all the sublists in an assistant.
---------------------------	--

Returns	string[]
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addSublist({
7   id : 'idname',
8   label : 'Sample label',
9   type : serverWidget.SublistType.LIST
10 });
11 var sublistid = assistant.getSublistIds();
12 ...
13 //Add additional code

```

Assistant.hasErrorHtml()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Indicates whether an assistant has an error message set. true if Assistant.errorHtml contains a value.
Returns	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({

```

```

4     title : 'Simple Assistant'
5 });
6 ...
7 if (assistant.hasErrorHtml()) {
8     ...
9 }
10 ...
11 //Add additional code

```

Assistant.isFinished()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Indicates whether all steps in an assistant are completed. If true, the assistant is finished and a completion message is displayed. To set the text for the completion message, use the Assistant.finishedHtml property.
Returns	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4     title : 'Simple Assistant'
5 });
6 ...
7 if (assistant.isFinished()) {
8     ...
9 }
10 ...
11 //Add additional code

```

Assistant.sendRedirect(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Manages redirects in an assistant. This method also addresses the case in which one assistant redirects to another assistant. In this scenario, the second assistant must return to the first assistant if the user Cancels or Finishes. This method, when used in the second assistant, ensures that users are redirected back to the first assistant.
---------------------------	--

Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.response	response	required	The response that redirects the user.	2015.2

Syntax

 Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/ui/serverWidget Module Script Samples .

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title: 'Small Business Setup Assistant',
5   hideNavBar: true
6 });
7 if (request.method === 'POST') {
8   if (assistant.getLastAction() === 'finish') {
9     assistant.finishedHtml = 'Completed!';
10    assistant.sendRedirect({
11      response: response
12    });
13  }
14 }
15 ...
16 //Add additional code

```

Assistant.setSplash(options)

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Method Description	Defines a splash message.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.title	string	required	The title of the splash screen.	2015.2
options.text1	string	required	Text for the splash screen	2015.2
options.text2	string	optional	Text for a second column on the splash screen, if desired.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/ui/serverWidget Module Script Samples .

```

1 ...
2 var assistant = serverWidget.createAssistant({
3   title : 'Simple Assistant'
4 });
5 assistant.setSplash({
6   title: "Welcome Title!",
7   text1: "An explanation of what this assistant accomplishes.",
8   text2: "Some parting words."
9 });
10 ...
11 //Add additional code

```

Assistant.updateDefaultValues(values)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Sets the default values of an array of fields that are specific to the assistant.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
values	Object[]	required	An object containing an array of name/value pairs that map field names to field values.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addField({
7   id : 'idname',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Sample label'
10});
11 assistant.updateDefaultValues({
12   idname : 'New Default Value'
13 });
14 ...
15 //Add additional code

```

Assistant.clientScriptFileDialog



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The File Cabinet ID of client script file to be used in this assistant.
Type	number
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You attempted to set this value when the Assistant.clientScriptModulePath property value has already been specified. For more information, see Assistant.clientScriptModulePath .

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.clientscriptId = 32;

```

```

7 | ...
8 | //Add additional code

```

Assistant.clientScriptModulePath

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The relative path to the client script file to be used in this assistant.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2016.2

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You attempted to set this value when the Assistant.clientScriptFileId property value has already been specified. For more information, see Assistant.clientScriptFileId .

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 | //Add additional code
2 | ...
3 | objAssistant.clientScriptModulePath = 'SuiteScripts/assistantBehavior.js';
4 | ...
5 | //Add additional code

```

Assistant.currentStep

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The current step. You can set any step as the current step.
Type	serverWidget.AssistantStep (read-only)
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var step = assistant.currentStep;
11 ...
12 //Add additional code

```

Assistant.errorHtml



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Error message text for the current step. Optionally, you can use HTML tags to format the message.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.errorHtml = "You have <b>not</b> filled out the required fields. Please go back.";
7 ...
8 //Add additional code

```

Assistant.finishedHtml



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The text to display after the assistant finishes. For example "You have completed the Small Business Setup Assistant. Take the rest of the day off".
-----------------------------	--

	To trigger display of the completion message, call Assistant.isFinished() .
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.finishedHtml = "Congratulations! You have successfully set up your account.";
7 ...
8 //Add additional code

```

Assistant.hideAddToShortcutsLink



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether to show or hide the Add to Shortcuts link that appears in the top-right corner of an assistant page. By default, the value is false, which means the Add to Shortcuts link is visible in the UI. If set to true, the Add To Shortcuts link is not visible on an Assistant page.
Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.hideAddToShortcutsLink = true;
7 ...
8 //Add additional code

```

Assistant.hideStepNumber



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether assistant steps are displayed with numbers. By default, the value is false, which means that steps are numbered. If set to true, the assistant does not use step numbers. To change step ordering, set Assistant.isNotOrdered .
Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 assistant.hideStepNumber = true;
11 ...
12 //Add additional code

```

Assistant.isNotOrdered



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether steps must be completed in a particular sequence. If steps are ordered, users must complete the current step before proceeding to the next step. The default value is false, which means the steps are ordered. Ordered steps appear vertically in the left panel of the assistant If set to true, steps can be completed in any order. In the UI, unordered steps appear horizontally and below the assistant title
Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 assistant.addStep({
11   id : 'idname2',
12   label : 'Sample label 2'
13 });
14 assistant.isNotOrdered = false;
15 ...
16 //Add additional code

```

Assistant.title

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The title for the assistant. The title appears at the top of all assistant pages. This value overrides the title specified in serverWidget.createAssistant(options) .
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var title = assistant.title;
7 ...
8 //Add additional code

```

serverWidget.AssistantStep

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A step within a custom NetSuite assistant.
---------------------------	--

	Create a step by calling Assistant.addStep(options) . For a complete list of this object's methods and properties, see AssistantStep Object Members .
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 ...
11 //Add additional code

```

AssistantStep.getFieldIds()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the IDs for all the fields in a step.
Returns	string[]
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',

```

```

8     label : 'Sample label'
9 });
10 assistant.addField({
11     id : 'fieldid',
12     type : serverWidget.FieldType.TEXT,
13     label : 'Field'
14 });
15 var fieldIds = assistantStep.getFieldIds();
16 ...
17 //Add additional code

```

AssistantStep.getLineCount(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the number of lines on a sublist in a step.
	Note: The first line number on a sublist is 0 (not 1).
Returns	number (the count of line items on a sublist)
	Note: if the sublist does not exist, -1 is returned.
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.group	string	required	The sublist internal ID.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4     title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7     id : 'idname',
8     label : 'Sample label'
9 });
10 var sublist = assistant.addSublist({

```

```

11 |     id : 'sublistid',
12 |     type : serverWidget.SublistType.INLINEEDITOR,
13 |     label : 'Editor'
14 |   });
15 |   var numLines = assistantStep.getLineCount({
16 |     group : 'sublistid'
17 |   });
18 |   ...
19 | //Add additional code

```

AssistantStep.getSublistFieldIds(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the IDs for all the sublist fields (line items) in a step.
Returns	string[]
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.group	string	required	The sublist internal ID.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var sublist = assistant.addSublist({
11   id : 'sublistid',
12   type : serverWidget.SublistType.INLINEEDITOR,
13   label : 'Editor'
14 });
15 var sublistFieldIds = assistantStep.getSublistFieldIds({
16   group : 'sublistid'
17 });
18 ...

```

```
19 //Add additional code
```

AssistantStep.getSublistValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the current value of a sublist field (line item) in a step.
Returns	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.group	string	required	The internal ID of the sublist.	2015.2
options.id	string	required	The internal ID of the sublist field.	2015.2
options.line	number	required	The line number for the sublist field.	2015.2



Note: The first line number on a sublist is 0 (not 1).

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```
1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',
8   label : 'Sample Label'
9 });
10 var sublist = assistant.addSublist({
11   id : 'sublistid',
12   type : serverWidget.SublistType.INLINEEDITOR,
13   label : 'Editor'
14 });
15 var sublistfield = sublist.addField({
16   id : 'fieldid',
17   type : serverWidget.FieldType.TEXT,
```

```

18     label : 'Text'
19 });
20 var sublistvalue = assistantStep.getSublistValue({
21   group: 'sublistid',
22   id: 'fieldid',
23   line: 0
24 });
25 ...
26 //Add additional code

```

AssistantStep.getSubmittedSublistIds()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the IDs for all the sublists submitted in a step.
Returns	string[]
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var sublist = assistant.addSublist({
11   id : 'sublistid',
12   type : serverWidget.SublistType.INLINEEDITOR,
13   label : 'Editor'
14 });
15 var submittedSublistId = assistantStep.getSubmittedSublistIds();
16 ...
17 //Add additional code

```

AssistantStep.getValue(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets the current value(s) of a field or multi-select field.
Returns	string (for standard fields) string[] (for multi-select fields)

Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of a field.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var field = assistant.addField({
11   id : 'fieldid',
12   type : serverWidget.FieldType.TEXT,
13   label : 'Text'
14 });
15 var value = assistantStep.getValue({
16   id: 'fieldid'
17 });
18 ...
19 //Add additional code

```

AssistantStep.helpText

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The help text for a step.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code ...
2 assistantStep.helpText = 'Help Text Goes Here.';
3 ...
4 //Add additional code

```

AssistantStep.id

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal ID of the step.
Type	string (read-only)
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var id = assistantStep.id';
11 ...
12 //Add additional code

```

AssistantStep.label

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The label for the step.
	Note: To create a label when the step is first added to the assistant, you can use the Assistant.addStep(options) method.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)

Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var label = assistantStep.label;
11 ...
12 //Add additional code

```

AssistantStep.stepNumber



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The numerical order of the step.
Type	number
<div style="background-color: #e0f2ff; padding: 5px;"> i Note: A sequence of assistant steps starts at 1. </div>	
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 var assistantStep = assistant.addStep({
7   id : 'idname',
8   label : 'Sample label'
9 });
10 var stepNum = assistantStep.stepNumber;
11 ...
12 //Add additional code

```

serverWidget.Button

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>A button that appears in a UI object.</p> <p>To add a button, use Form.addButton(options) or Sublist.addButton(options). When adding a button to a record or form, consider using a beforeLoad user event script.</p> <p>Custom buttons only appear during Edit mode. On records, custom buttons appear to the left of the printer icon.</p> <p>Note: Currently you cannot use SuiteScript to add or remove a custom button to or from the More Actions menu. You can, however, do this using SuiteBuilder point-and-click customization. See the help topic Configuring Buttons and Actions.</p> <p>For a complete list of this object's properties, see Button Object Members.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var button = form.addButton({
7   id : 'buttonid',
8   label : 'Test'
9 });
10 ...
11 //Add additional code
12

```

Button.isDisabled

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>Whether a button is grayed-out and disabled.</p> <p>The default value is <code>false</code>.</p> <p>If set to true, the button appears grayed-out in the and cannot be clicked.</p> <p>Note: This method is not supported for standard NetSuite buttons. This method can be used with custom buttons only.</p>
Type	boolean

Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var button = form.addButton({
7   id : 'buttonid',
8   label : 'Test'
9 });
10 button.isDisabled = true;
11 ...
12 //Add additional code

```

Button.isHidden

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Whether the button is hidden in the UI. The default value is false, which means the button is visible. If set to true, the button is not visible in the UI.
Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });

```

```

6  var button = form.addButton({
7      id : 'buttonid',
8      label : 'Test'
9  });
10 button.isHidden = true;
11 ...
12 //Add additional code

```

Button.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The label for the button. You can use this property to rename a button based on context, for example to re-label a button for particular users that are viewing a page. Note: This property is supported on custom buttons and most standard buttons.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4     title : 'Simple Form'
5 });
6 var button = form.addButton({
7     id : 'buttonid',
8     label : 'Test'
9 });
10 var label = button.label;
11 ...
12 //Add additional code

```

serverWidget.Field



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A body or sublist field. Use fields to record or display information specific to your needs. To add a Field object, use Assistant.addField(options) , Form.addField(options) , or Sublist.addField(options) . For a complete list of this object's methods and properties, see Field Object Members .
---------------------------	---

Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_text',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 ...
12 //Add additional code

```

Field.addSelectOption(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds the select options that appears in the dropdown of a field.
	 Important: After you create a select or multi-select field that is sourced from a record or list, you cannot add additional values with Field.addSelectOption(options). The select values are determined by the source record or list.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.value	string	required	The internal ID of this select option.	2015.2
options.text	string	required	The label for this select option.	2015.2

Parameter	Type	Required / Optional	Description	Since
options.isSelected	boolean	optional	If set to true, this option is selected by default in the UI. The default value for this parameter is false.	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var selectField = form.addField({
7   id : 'custpage_selectfield',
8   type : serverWidget.FieldType.SELECT,
9   label : 'Select'
10 });
11 selectField.addSelectOption({
12   value : '',
13   text : ''
14 });
15 selectField.addSelectOption({
16   value : 'a',
17   text : 'Albert'
18 });
19 ...
20
21 //Add additional code
22

```

Field.getSelectOptions(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	<p>Obtains a list of available options on a select field.</p> <p>The internal ID and label of the options for a select field as name/value pairs is returned.</p> <p>The first 1,000 available options are returned.</p> <p>If you attempt to get select options on a field that is not a select field, if it is a popup select field, or if you reference a field that does not exist on the form, null is returned.</p> <p>i Note: A call to this method may return different results for the same field for different roles.</p>
Returns	Object[]
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.filter	string	optional	A search string to filter the select options that are returned. For example, if there are 50 select options available, and 10 of the options contains 'John', e.g. "John Smith" or "Shauna Johnson", only those 10 options will be returned. Filter values are case insensitive. The filters 'John' and 'john' will return the same select options.	2015.2
options.filteroperator	string	optional	Supported operators are contains is startswith. If not specified, defaults to the contains operator.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var selectField = form.addField({
7   id : 'custpage_selectfield',
8   type : serverWidget.FieldType.SELECT,
9   label : 'Select'
10 });
11 selectField.addSelectOption({
12   value : 'a',
13   text : 'Albert'
14 });
15 var options = field.getSelectOptions({
16   filter : 'a',
17   filteroperator: 'startswith'
18 });
19 ...
20 //Add additional code

```

Field.setHelpText(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the help text for the field. When the field label is clicked, a popup displays the help text defined using this method.
Returns	The <code>serverWidget.Field</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None

Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.help	string	required	The text in the field help popup.	2015.2
options.showInlineForAssistant	boolean	optional	If set to true, the field help will display inline below the field on the assistant, and in a field help popup. The default value is false, which means the field help appears in a popup when the field label is clicked and does not appear inline.	2015.2

Note: The inline parameter is available only to `serverWidget.Field` objects that have been added to `serverWidget.Assistant` objects.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/ui/serverWidget Module Script Samples .

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.setHelpText({
12   help : "Help Text Goes Here."
13 });
14 ...
15 //Add additional code

```

Field.updateBreakType(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Updates the break type of the field. Set this value using the <code>FieldBreakType</code> enum. The break type is used to add a break in flow layout for the field.
Returns	<code>serverWidget.Field</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)

Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.breakType	string	required	The break type of the field. Set this value using the serverWidget.FieldBreakType enum	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.updateBreakType({
12   breakType : serverWidget.FieldBreakType.STARTCOL
13 });
14 ...
15 //Add additional code

```

Field.updateDisplaySize(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Updates the width and height of the field. Only supported on multi-selects, long text, and fields that get rendered as INPUT (type=text) fields. This function is not supported on list/record fields or rich text fields. To set height and width for rich text fields, use Field.richTextWidth and Field.richTextHeight .
Returns	serverWidget.Field object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.height	number	required	The new height of the field.	2015.2
options.width	number	required	The new width of the field.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.updateDisplaySize({
12   height : 60,
13   width : 100
14 });
15 ...
16 //Add additional code

```

Field.updateDisplayType(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Updates the display type for the field.
Returns	serverWidget.Field object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.displayType	string	required	The new display type of the field. For more information about possible values, see serverWidget.FieldDisplayType .	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.updateDisplayType({
12   displayType : serverWidget.FieldDisplayType.HIDDEN
13 });
14 ...
15 //Add additional code

```

Field.updateLayoutType(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Updates the layout type for the field.
Returns	serverWidget.Field object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.layoutType	string	required	The new layout type of the field. Set this value using the serverWidget.FieldLayoutType enum.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code

```

```

2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.updateLayoutType({
12   layoutType : serverWidget.FieldLayoutType.NORMAL
13 });
14 ...
15 //Add additional code

```

Field.alias



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	An alternate name that you can assign to a serverWidget.Field object. By default, the alias is equal to the field's internal ID. This property is only supported on scripted fields created using the N/ui/serverWidget Module .
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.alias = 'fieldid';
12 ...
13 //Add additional code

```

Field.defaultValue



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The default value for this field.
-----------------------------	-----------------------------------

	If you pass an empty string or any value that is not a number, such as undefined, the field defaults to a blank field in the UI. This property is supported only on scripted fields created using the N/ui/serverWidget Module .
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.defaultValue = 'Insert Text Here.';
12 ...
13 //Add additional code

```

Field.helpText

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The help text for the field.
Type	string (read-only)
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	-

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'

```

```

5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.setHelpText({
12   help : "Help Text Goes Here."
13 });
14 var fieldHelpText = field.helpText;
15 ...
16 //Add additional code

```

Field.id



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The field internal ID.
Type	string (read-only)
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 var fieldId = field.id;
12 ...
13 //Add additional code

```

Field.isMandatory



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>Whether the field is mandatory or optional.</p> <p>If set to true, then the field is defined as mandatory.</p> <p>The default value is false.</p> <p>This property is supported only on scripted fields created using the N/ui/serverWidget Module.</p>
-----------------------------	--

Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.isMandatory = true;
12 ...
13 //Add additional code

```

Field.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The field label. There is a 40-character limit for custom field labels.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',

```

```

8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 var label = field.label;
12 ...
13 //Add additional code

```

Field.linkText

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The text displayed for a link in place of the URL.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.URL,
9   label : 'URL'
10 });
11 field.linkText = 'NetSuite';
12 ...
13 //Add additional code

```

Field.maxLength

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The maximum length, in characters, of the field (only valid for text, rich text, long text, and textarea fields). This property is supported only on scripted fields created using the N/ui/serverWidget Module .
Type	number
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module

Since	2015.2
-------	--------

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.maxLength = 64;
12 ...
13 //Add additional code

```

Field.padding

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The number of empty vertical character spaces above the field. This property is supported only on scripted fields created using the N/ui/serverWidget Module .
Type	number
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.padding = 50;
12 ...
13 //Add additional code

```

Field.richTextHeight



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The height of a rich text field, in pixels. The minimum value is 100 pixels and the maximum value is 500 pixels.
	Note: Rich Text Editing must be enabled.
Type	number
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.RICHTEXT,
9   label : 'Rich Text'
10 });
11 field.richTextHeight = 50;
12 ...
13 //Add additional code

```

Field.richTextWidth



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The width of a rich text field, in pixels. The minimum value is 250 pixels and the maximum value is 800 pixels.
	Note: Rich Text Editing must be enabled.
Type	number
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module

Since	2015.2
-------	--------

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.RICHTEXT,
9   label : 'Rich Text'
10 });
11 field.richTextWidth = 100;
12 ...
13 //Add additional code

```

Field.type



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns the type of a body or sublist field. For example, the value can return text, date, currency, select, checkbox, etc. For more information on possible return values, see format.Type . The maximum character limit for select field types is 801.
Type	string (read-only)
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_textfield',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 var fieldtype = field.type;
12 ...
13 //Add additional code

```

serverWidget.FieldGroup



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A field group on serverWidget.Assistant objects and on serverWidget.Form objects. A field group is a collection of fields that can be displayed in a one or two column format. Assign a field to a field group in order to label, hide or collapse a group of fields. For a complete list of this object's properties, see FieldGroup Object Members .
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var fieldgroup = form.addFieldGroup({
7   id : 'fieldgroupid',
8   label : 'Field Group'
9 });
10 var field = form.addField({
11   id : 'custpage_textfield',
12   type : serverWidget.FieldType.TEXT,
13   label : 'Text',
14   container : 'fieldgroupid'
15 });
16 ...
17 //Add additional code

```

FieldGroup.isBorderHidden



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the field group border is hidden. If set to false, a border around the field group is displayed. The default value is false.
Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var fieldgroup = form.addFieldGroup({
7   id : 'fieldgroupid',
8   label : 'Field Group'
9 });
10 var field = form.addField({
11   id : 'custpage_textfield',
12   type : serverWidget.FieldType.TEXT,
13   label : 'Text',
14   container : 'fieldgroupid'
15 });
16 fieldgroup.isBorderHidden = true;
17 ...
18 //Add additional code

```

FieldGroup.isCollapsible



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the field group can be collapsed. The default value is false, which means the field group displays as a static group that cannot be opened or closed. If set to true, the field group can be collapsed. Only supported for fields on serverWidget.Assistant objects
Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var fieldgroup = form.addFieldGroup({
7   id : 'fieldgroupid',
8   label : 'Field Group'
9 });
10 var field = form.addField({
11   id : 'custpage_textfield',
12   type : serverWidget.FieldType.TEXT,
13   label : 'Text',
14   container : 'fieldgroupid'
15 });
16 fieldgroup.isBorderHidden = true;
17 ...
18 //Add additional code

```

```

12     type : serverWidget.FieldType.TEXT,
13     label : 'Text',
14     container : 'fieldgroupid'
15   });
16   fieldgroup.isCollapsible = true;
17 ...
18 //Add additional code

```

FieldGroup.isCollapsed



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether field group is collapsed or expanded. The default value is false, which means that when the page loads, the field group will not appear collapsed. If set to true, the field group is collapsed. Only supported for fields on serverWidget.Assistant objects
Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var fieldgroup = form.addFieldGroup({
7   id : 'fieldgroupid',
8   label : 'Field Group'
9 });
10 var field = form.addField({
11   id : 'custpage_textfield',
12   type : serverWidget.FieldType.TEXT,
13   label : 'Text',
14   container : 'fieldgroupid'
15 });
16 var collapsed = fieldgroup.isCollapsed;
17 ...
18 //Add additional code

```

FieldGroup.isSingleColumn



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Indicates whether the field group is displayed into a single column. The default value is false.
-----------------------------	---

Type	boolean
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4     title : 'Simple Form'
5 });
6 var fieldgroup = form.addFieldGroup({
7     id : 'fieldgroupid',
8     label : 'Field Group'
9 });
10 var field = form.addField({
11     id : 'custpage_textfield',
12     type : serverWidget.FieldType.TEXT,
13     label : 'Text',
14     container : 'fieldgroupid'
15 });
16 var aligned = fieldgroup.isSingleColumn;
17 ...
18 //Add additional code

```

FieldGroup.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The label for the field group.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4     title : 'Simple Form'
5 });
6 var fieldgroup = form.addFieldGroup({
7     id : 'fieldgroupid',
8     label : 'Field Group'

```

```

9 });
10 var field = form.addField({
11   id : 'custpage_textfield',
12   type : serverWidget.FieldType.TEXT,
13   label : 'Text',
14   container : 'fieldgroupid'
15 });
16 var label = fieldgroup.label;
17 ...
18 //Add additional code

```

serverWidget.Form

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A NetSuite-looking form After you create a Form object, you can: <ul style="list-style-type: none">■ Add a variety of scriptable elements to the form including fields, links, buttons, tabs, and sublists. For a complete list of this object's methods and properties, see Form Object Members .
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>(beforeLoad(scriptContext))</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 ...
7 //Add additional code

```

Form.addButton(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a button to a form.
Returns	serverWidget.Button object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>(beforeLoad(scriptContext))</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of the button. If you are adding the button to an existing page, the internal ID must be in lowercase, contain no spaces, and include the prefix custpage. For example, if you add a button that appears as Update Order , the button internal ID should be something similar to custpage_updateorder .	2015.2
options.label	string	required	The label for this button.	2015.2
options.functionName	string	optional	The function name to be triggered on a click event.	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6
7 form.addButton({
8   id : 'buttonid',
9   label : 'Test'
10 });
11 ...
12 //Add additional code

```

Form.addCredentialField(options)

Note: The content in this help topic pertains to SuiteScript 2.0.	
Method Description	<p>Adds a text field that lets you store credentials in NetSuite to be used when invoking services provided by third parties. The GUID generated by this field can be reused multiple times until the script executes again.</p> <p>For example, when executing credit card transactions, merchants need to store credentials in NetSuite that are used to communicate with Payment Gateway providers.</p> <p>The credentials added with this method can be used with the N/sftp Module and the N/https Module.</p> <p>Note the following about this method:</p> <ul style="list-style-type: none"> ▪ Credentials associated with this field are stored in encrypted form. ▪ No piece of SuiteScript holds a credential in clear text mode. ▪ NetSuite reports or forms will never provide to the end user the clear text form of a credential. ▪ Any exchange of the clear text version of a credential with a third party must occur over SSL.

	<ul style="list-style-type: none"> ■ For no reason will NetSuite ever log the clear text value of a credential (for example, errors, debug message, alerts, system notes, and so on). ■ Decryption occurs through the scripts listed in the <code>restrictToScriptIds</code> parameter. These scripts can call https.createSecureString(options) to decrypt the GUID and create a <code>SecureString</code> instance. <div style="border: 1px solid #f0e68c; padding: 5px; margin-top: 10px;"> ⚠ Important: The default maximum length for a credential field is 32 characters. If needed, use the <code>Field.maxLength</code> property to change this value. </div>
Returns	<code>serverWidget.Field</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
<code>options.id</code>	string	required	The internal ID of the credential field. The internal ID must be in lowercase, contain no spaces, and include the prefix <code>custpage</code> if you are adding the field to an existing page.	2015.2
<code>options.label</code>	string	required	The label for the credential field.	2015.2
<code>options.restrictToDomains</code>	string string[]	required	The domains that the credentials can be sent to, such as 'www.mysite.com'. Credentials cannot be sent to a domain that is not specified here. This value can be a domain or a list of domains to which the credentials can be sent.	2015.2
<code>options.restrictToScriptIds</code>	string string[]	required	The IDs of the scripts that are allowed to use this credential field. For example, 'customscript_my_script'. Scripts defined here can call https.createSecureString(options) to decrypt the GUID.	2015.2
<code>options.restrictToCurrentUser</code>	boolean	optional	Controls whether use of this credential is restricted to the same user that originally entered the credential. By default, the value is <code>false</code> , which means that multiple users can use the credential. For example, multiple clerks at a store making secure calls to a credit processor using a credential that represents the company they work for. If set to <code>true</code> , the credentials apply to a single user.	2015.2
<code>options.container</code>	string	optional	The internal ID of the tab or field group to add the credential field to. By default, the field is added to the main section of the form.	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete N/ui/serverWidget Module script example, see [N/ui/serverWidget Module Script Samples](#). For a complete script sample that uses Form.addCredentialField, see [N/https Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var credField = form.addCredentialField({
7   id : 'username',
8   label : 'Username',
9   restrictToDomains : 'www.mysite.com',
10  restrictToScriptIds : 'customscript_my_script',
11  restrictToCurrentUser : false,
12 });
13 credField.maxLength = 64;
14 ...
15 //Add additional code

```

Form.addField(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a field to a form.
Returns	serverWidget.Field object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of the field. The internal ID must be in lowercase, contain no spaces, and include the prefix custpage if you are adding the field to an existing page. For example, if you add a field that appears as Purchase Details , the field internal ID should be something similar to custpage_purchasedetails or custpage_purchase_details.	2015.2
options.label	string	required	The label for this field.	2015.2
options.type	string	required	The field type for the field. Use the serverWidget.FieldType enum to define the field type.	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p>Important: Long text fields created with SuiteScript have a character limit of 100,000. Long text fields created with Suitebuilder have a character limit of 1,000,000.</p>	
options.source	string	optional	<p>The internalId or scriptId of the source list for this field if it is a select (List/Record) or multi-select field.</p> <p>Important: After you create a select or multi-select field that is sourced from a record or list, you cannot add additional values with <code>Field.addSelectOption(options)</code>. The select values are determined by the source record or list.</p> <p>Note: For radio fields only, the <code>source</code> parameter must contain the internal ID for the field. For more information about working with radio buttons, see the help topic Working with Radio Buttons.</p>	2015.2
options.container	string	optional	<p>The internal ID of the tab or field group to add the field to.</p> <p>By default, the field is added to the main section of the form.</p>	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_abc_text',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 ...
12 //Add additional code

```

Form.addFieldGroup(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a group of fields to a form.
Returns	<code>serverWidget.FieldGroup</code> object

Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.id</code>	string	required	An internal ID for the field group.	2015.2
<code>options.label</code>	string	required	The label for this field group.	2015.2
<code>options.tab</code>	string	optional	The internal ID of the tab to add the field group to. By default, the field group is added to the main section of the form.	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var fieldgroup = form.addFieldGroup({
7   id : 'fieldgroupid',
8   label : 'Field Group'
9 });
10 var field = form.addField({
11   id : 'custpage_text',
12   type : serverWidget.FieldType.TEXT,
13   label : 'Text',
14   container : 'fieldgroupid'
15 });
16 ...
17 //Add additional code

```

Form.addPageInitMessage(options)

Method Description	Shows a message when users view a record or Suitelet. User event context can be used to control whether the message is shown on records in view, create, or edit mode (not applicable for Suitelets). See the help topic context.UserEventType . This method is called during a <code>beforeLoad</code> user event or in a Suitelet, and the message is later displayed on the client side, after the <code>pageInit</code> script is completed.
Returns	<code>void</code>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)

Governance	None
Module	N/ui/serverWidget Module
Since	2018.2

Parameters

The options object passed to the `Form.addPageInitMessage(options)` method takes a single property; either a `message.Message` object, or the same options object that can be passed to the `message.create(options)` method. The following tables list the parameters for the previously mentioned object property possibilities.

The `message.Message` object and the `message.Type` enum require loading the [N/ui/message Module](#).

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
<code>options.message</code>	<code>message.Message</code>	required	Encapsulates the message to be shown on the form.	2018.2

Parameter	Type	Required / Optional	Description	Since
<code>options.type</code>	<code>message.Type</code>	required	The message type.	2016.1
<code>options.title</code>	string	optional	The message title. This value defaults to an empty string.	2016.1
<code>options.message</code>	string	optional	The content of the message. This value defaults to an empty string.	2016.1
<code>options.duration</code>	int	optional	<p>The amount of time, in milliseconds, to show the message. The default is 0, which shows the message until <code>Message.hide()</code> is called.</p> <p>If you specify a duration for <code>message.create()</code> and <code>message.show()</code>, the value from the <code>message.show()</code> method call takes precedence.</p>	2018.2

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/ui/serverWidget Module Script Samples .
---	---

```

1 //Add additional code
2 ...
3
4 // Options object as parameter
5 form.addPageInitMessage({type: message.Type.INFORMATION, message: 'Hello world!', duration: 5000});
6
7 // Message object as parameter
8 var messageObj = message.create({type: message.Type.INFORMATION, message: 'Hello world!', duration: 5000}); form.addPageInitMes
9 sage({message: messageObj});
10 // Show message when the record is in view mode

```

```

11 function beforeLoad(context) {
12     if(context.type === 'view')
13         context.form.addPageInitMessage(messageOptions);
14 }
15 ...
16 //Add additional code

```

Form.addPageLink(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a link to a form.
	i Note: You cannot choose where the page link appears.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	required	The text label for the link.	2015.2
options.type	string	required	The type of page link to add. Use the serverWidget.FormPageLinkType enum to set the value.	2015.2
options.url	string	required	The URL for the link.	2015.2

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4     title : 'Simple Form'
5 });
6 form.addPageLink({
7     type : serverWidget.FormPageLinkType.CROSSLINK,
8     title : 'NetSuite',
9     url : 'http://www.netsuite.com'
10 })
11 ...
12 //Add additional code

```

Form.addButton(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a reset button to a form. The reset buttons enables a user to clear the entries.
Returns	serverWidget.Button object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.label	string	optional	The label used for this button. If no label is provided, the label defaults to Reset .	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addButton({
7   label : 'Reset Button'
8 });
9 ...
10 //Add additional code

```

Form.addSecretKeyField(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a secret key field to the form. This key can be used in crypto modules to perform encryption or hashing.
Returns	serverWidget.Field object



Important: The default maximum length for a secret key field is 32 characters. If needed, use the [Field.maxLength](#) property to change this value.

Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of the secret key field. The internal ID must be in lowercase, contain no spaces, and include the prefix custpage if you are adding the field to an existing page.	2016.1
options.restrictToScriptIds	string string[]	required	The ID or list of IDs of the scripts where the key can be used.	2016.1
options.label	string	required	The UI label for the field.	2016.1
options.restrictToCurrentUser	boolean	optional	Controls whether use of this secret key is restricted to the same user that originally entered the key. By default, the value is false, which means that multiple users can use the key. If set to true, the secret key applies to a single user.	2016.1
options.container	string	optional	The internal ID of the tab or field group to add the field to. By default, the field is added to the main section of the form.	2016.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete N/ui/serverWidget Module script example, see [N/ui/serverWidget Module Script Samples](#). For a complete script example that uses Form.addSecretKeyField(options), see [N/crypto Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 skField = form.addSecretKeyField({
7   id : 'password',
8   restrictToScriptIds : 'customscript_my_script',
9   label : 'Password',
10  restrictToCurrentUser : false
11 });

```

```

12 | skField.maxLength = 64;
13 | ...
14 | //Add additional code

```

Form.addSublist(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Add a sublist to a form.
	 Note: If the row count exceeds 25, sorting is not supported on static sublists created using this method.
Returns	A serverWidget.Sublist object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID name of the sublist. The internal ID must be in lowercase, contain no spaces, and include the prefix custpage if you are adding the sublist to an existing page. For example, if you add a sublist that appears as Purchase Details , the sublist internal ID should be something equivalent to custpage_purchasedetails or custpage_purchase_details.	2015.2
options.label	string	required	The label for this sublist.	2015.2
options.tab	string	optional	The tab under which to display this sublist. If empty, the sublist is added to the main tab.	2015.2
options.type	string	required	The sublist type. Use the serverWidget.SublistType enum to set the value.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 | //Add additional code

```

```

2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublistid',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 ...
12 //Add additional code

```

Form.addButton(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a submit button to a form.
	Note: If the row count exceeds 25, sorting is not supported on static sublists created using this method.
Returns	serverWidget.Button object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.label	string	optional	The label for this button. If no label is provided, the label defaults to "Save".	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addButton({
7   label : 'Submit Button'
8 });
9 ...
10 //Add additional code

```

Form.addSubtab(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a subtab to a form.
	<p>Note: In order for your subtab to appear on your form, there must be at least one object assigned to the subtab. Otherwise, the subtab will not appear.</p>
	<p>Note: If you have less than two subtabs on your form, the subtab will not appear. Instead the fields assigned to the tab will appear at the bottom of the form.</p>
Returns	serverWidget.Tab object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID name of the subtab. The internal ID must be in lowercase, contain no spaces. If you are adding the subtab to an existing page, include the prefix <code>custpage</code> . For example, if you add a subtab that appears as Purchase Details , the subtab internal ID should be something similar to <code>custpage_purchasedetails</code> or <code>custpage_purchase_details</code> .	2015.2
options.label	string	required	The label for this subtab.	2015.2
options.tab	string	optional	The tab under which to display this sublist. If empty, the sublist is added to the main tab.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addSubtab({
7   id : 'subtabid',
8   label : 'Subtab'
```

```

9 });
10 ...
11 //Add additional code

```

Form.addTab(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a tab to a form.
	<p>i Note: In order for your tab to appear on your form, there must be at least one object assigned to the tab. Otherwise, the tab will not appear.</p>
	<p>i Note: If you have less than two tabs on your form, the tab will not appear. Instead the fields assigned to the tab will appear at the bottom of the form.</p>
Returns	serverWidget.Tab object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID name of the tab. The internal ID must be in lowercase and contain no spaces. If you are adding the tab to an existing page, include the prefix custpage. For example, if you add a subtab that appears as Purchase Details , the subtab internal ID should be something similar to custpage_purchasedetails or custpage_purchase_details.	2015.2
options.label	string	required	The label for this tab.	2015.2

Syntax

! **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var tab = form.addTab({
7   id : 'tabid',

```

```

8   label : 'Tab'
9 });
10 ...
11 //Add additional code

```

Form.getButton(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a Button object by internal ID.
Returns	<code>serverWidget.Button</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID name of the button. Internal IDs must be in lowercase and contain no spaces.	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var button = form.addButton({
7   id : 'buttonid',
8   label : 'Test'
9 });
10 var button = form.getButton({
11   id : 'buttonid'
12 });
13 ...
14 //Add additional code

```

Form.getField(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a Field object by internal ID.
---------------------------	--

Returns	serverWidget.Field object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID name of the field. Internal IDs must be in lowercase and contain no spaces.	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addField({
7   id : 'custpage_text',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10});
11 var field = form.getField({
12   id : 'textfield'
13});
14 ...
15 //Add additional code

```

Form.getSublist(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a Sublist object by internal ID.
Returns	serverWidget.Sublist object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID name of the sublist. Internal IDs must be in lowercase and contain no spaces.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addSublist({
7   id : 'sublistid',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 var sublist = form.getSublist({
12   id : 'sublistid'
13 });
14 ...
15 //Add additional code

```

Form.getSubtab(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a subtab by internal ID.
Returns	<code>serverWidget.Tab</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID name of the subtab.	2015.2

Parameter	Type	Required / Optional	Description	Since
			Internal IDs must be in lowercase and contain no spaces.	

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addSubtab({
7   id : 'subtabid',
8   label : 'Subtab'
9 });
10 var subtab = form.getSubtab({
11   id : 'subtabid'
12 });
13 ...
14 //Add additional code

```

Form.getTab(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a tab object from its internal ID.
Returns	<code>serverWidget.Tab</code> object.
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.id</code>	string	required	The internal ID name of the tab to retrieve.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```
1 //Add additional code
```

```

2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addTab({
7   id : 'tabid',
8   label : 'Tab'
9 });
10 var tab = form.getTab({
11   id : 'tabid'
12 });
13 ...
14 //Add additional code

```

Form.getTabs()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an array that contains all the tabs in a form.
Returns	<code>serverWidget.Tab[]</code> objects
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addTab({
7   id : 'tabid',
8   label : 'Tab'
9 });
10 var tabs = form.getTabs();
11 ...
12 //Add additional code

```

Form.insertField(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Inserts a field in front of another field.
Returns	<code>void</code>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))

Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.field	serverWidget.Field	required	The Field object to insert.	2016.1
options.nextfield	string	required	The internal ID name of the field you are inserting a field in front of.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field1 = form.addField({
7   id : 'custpage_text',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 var field2 = form.addField({
12   id : 'custpage_text2',
13   type : serverWidget.FieldType.TEXT,
14   label : 'Text'
15 });
16 form.insertField({
17   field : field2,
18   nextfield : 'textfield1'
19 });
20 ...
21 //Add additional code

```

Form.insertSublist(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Inserts a sublist in front of another sublist.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None

Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.sublist	serverWidget.Sublist	required	The Sublist object to insert.	2016.1
options.nextsublist	string	required	The internal ID name of the sublist you are inserting a sublist in front of.	2015.2

Syntax

 Important:	The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/ui/serverWidget Module Script Samples .
---	---

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist1 = form.addSublist({
7   id : 'sublistid1',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 var sublist2 = form.addSublist({
12   id : 'sublistid2',
13   type : serverWidget.SublistType.INLINEEDITOR,
14   label : 'Inline Editor Sublist'
15 });
16 form.insertSublist({
17   sublist : sublist2,
18   nextsublist : 'sublistid1'
19 });
20 ...
21 //Add additional code

```

Form.insertSubtab(options)

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Method Description	Inserts a subtab in front of another subtab.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module

Since	2015.2
--------------	--------

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.subtab	serverWidget.Tab	required	The Subtab object to insert.	2016.1
options.nextsubtab	string	required	The internal ID name of the subtab you are inserting a subtab in front of.	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var subtab1 = form.addSubtab({
7   id : 'subtabid1',
8   label : 'Subtab'
9 });
10 var subtab2 = form.addSubtab({
11   id : 'subtabid2',
12   label : 'Subtab'
13 });
14 form.insertSubtab({
15   subtab : subtab2,
16   nextsubtab : 'subtabid1'
17 });
18 ...
19 //Add additional code

```

Form.insertTab(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Inserts a tab in front of another tab.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.tab	serverWidget.Tab	required	The Tab object to insert.	2016.1
options.nexttab	string	required	The internal ID name of the tab you are inserting a tab in front of.	2015.2

Syntax

 Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/ui/serverWidget Module Script Samples .

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var tab1 = form.addTab({
7   id : 'tabid1',
8   label : 'Tab'
9 });
10 var tab2 = form.addTab({
11   id : 'tabid2',
12   label : 'Tab'
13 });
14 form.insertTab({
15   tab: tab2,
16   nexttab:'tabid1'
17 });
18 ...
19 //Add additional code

```

Form.removeButton(options)

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Method Description	Removes a button. This method can be used on custom buttons and certain built-in NetSuite buttons. For more information about built-in NetSuite buttons, see the help topic Button IDs .
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext)) For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID name of the button to remove. See the help topic Button IDs . The internal ID must be in lowercase and contain no spaces.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 function beforeLoad(context) {
4     var yourForm = context.form;
5     yourForm.removeButton('refund');
6 }
7 ...
8 //Add additional code

```

Form.updateDefaultValues(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Updates the default values of multiple fields on the form.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
values	Object[]	required	An object containing an array of name/value pairs that map field names to field values.	2016.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```
1 //Add additional code
```

```

2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_text',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 form.updateDefaultValues({
12   custpage_text : 'Text Goes Here'
13 })
14 ...
15 //Add additional code

```

Form.clientScriptFileId

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The internal file ID of client script file to be used in this form. Use this property when attaching an on demand client script to a server-side script.
Type	number
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You attempted to set this value when the Form.clientScriptModulePath property value has already been specified. For more information, see Form.clientScriptModulePath .

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.clientScriptFileId = 32;
7 ...
8 //Add additional code

```

Form.clientScriptModulePath



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The relative path to the client script file to be used in this form. Use this property when attaching an on demand client script to a server-side script.
	Note: If you deploy a client script to a form using Form.clientScriptFileId or Form.clientScriptModulePath, using the N/log Module adds the logs to the deployment of the parent script. The parent script can be either a beforeLoad user event script or a SuiteScript 2.0 Suitelet Script Type .
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2016.2

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You attempted to set this value when the Form.clientScriptFileId property value has already been specified. For more information, see Form.clientScriptFileId .

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 objForm.clientScriptModulePath = 'SuiteScripts/formBehavior.js';
4 ...
5 //Add additional code

```

Form.title



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The title used for the form.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var title = form.title;
7 ...
8 //Add additional code

```

serverWidget.List



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A list page. For a complete list of this object's methods and properties, see List Object Members .
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 ...
7 //Add additional code

```

List.addButton(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a button to a list.
Returns	<code>serverWidget.Button</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module

Since	2015.2
-------	--------

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of the button. The internal ID must be in lowercase, contain no spaces, and include the prefix <code>custpage</code> if you are adding the button to an existing page. For example, if you add a button that appears as Update Order , the button internal ID should be something similar to <code>custpage_updateorder</code> .	2015.2
options.label	string	required	The label for this button.	2015.2
options.functionName	string	optional	The function name to call when clicking on this button.	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 list.addButton({
7   id : 'buttonid',
8   label : 'Test'
9 });
10 ...
11 //Add additional code

```

List.addColumn(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a column to a list.
Returns	<code>serverWidget.ListColumn</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.				
Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID of this column. The internal ID must be in lowercase, contain no spaces.	2015.2
options.label	string	required	The label for this column.	2015.2
options.type	string	required	The field type for this column. Set this value using the serverWidget.FieldType enum. Note: The CHECKBOX field type is not supported.	2015.2
options.align	string	optional	The layout justification for this column. Set this value using the serverWidget.LayoutJustification enum. The default value is LEFT.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 list.addColumn({
7   id : 'column1',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text',
10  align : serverWidget.LayoutJustification.RIGHT
11 });
12 ...
13 //Add additional code

```

List.addEditColumn(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a column containing Edit or Edit/View links to a Suitelet or Portlet list. These Edit or Edit/View links appear to the left of a previously existing column.
Returns	serverWidget.ListColumn object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))

Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description	Since
options.column	serverWidget.ListColumn	required	Column that acts as a reference. The Edit/View column is added to the left of the column specified here.	2015.2
options.showHrefCol	boolean	optional	If set to true, the URL for the link is clickable.	2015.2
options.showView	boolean	optional	If true then an Edit/View column will be added. Otherwise only an Edit column will be added.	2015.2
options.link	string	optional	The Edit/View base link. (For example: /app/common/entity/employee.nl) The complete link is formed like this: <link>?<linkParamName>=<row data from linkParam>. (For example: /app/common/entity/employee.nl?id=123)	2019.2
options.linkParam	string	optional	The internal ID of the field in the row data where to take the parameter from. The default value is the value set in the options.column parameter.	2019.2
options.linkParamName	string	optional	 Tip: In most cases, the value to use here is internalid The name of the parameter. The default value is id.	2019.2

Syntax

 Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see N/ui/serverWidget Module Script Samples .

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6
7 list.addColumn({
8   id: 'internalid',
9   type: 'text',
10  label: 'Number'
11 });
12 list.addColumn({
13   id: 'entityid',
14   type: 'text',

```

```

15     label: 'Name'
16 });
17 list.addEditColumn({
18     column : 'entityid',
19     showHrefCol: true,
20     showView : true,
21     link: '/app/common/entity/employee.nl',
22     linkParam: 'internalid',
23     linkParamName: 'id',
24 });
25 ...
26 //Add additional code

```

List.addPageLink(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a link to a list.
Returns	serverWidget.List (it return itself)
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	required	The text label for the link.	2015.2
options.type	string	required	The type of page link to add. Set this value using the serverWidget.FormPageLinkType enum.	2015.2
options.url	string	required	The URL for the link.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4     title : 'Simple List'
5 });
6 list.addPageLink({
7     title : 'NetSuite',
8     type : serverWidget.FormPageLinkType.CROSSLINK,
9     url : 'http://www.netsuite.com'
10 });

```

```

11 ...
12 //Add additional code

```

List.addRow(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a single row to a list.
Returns	serverWidget.List
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.row</code>	<code>Object[]</code>	required	A row that consists of either a search.Result array, or name/value pairs. Each pair should contain the value for the corresponding Column object in the list.	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 list.addRow({
7   row : { columnid1 : 'value1', columnid2 : 'value2' }
8 });
9 ...
10 //Add additional code

```

List.addRows(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds multiple rows to a list.
Returns	serverWidget.List

Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.rows</code>	Object[]	required	An array of rows that consist of either a <code>search.Result</code> array, or an array of name/value pairs. Each pair should contain the value for the corresponding Column object in the list.	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 list.addRows({
4     rows : [{columnid1 : 'value1', columnid2 : 'value2'},
5             {columnid1 : 'value2', columnid2 : 'value3'}]
6 });
7 ...
8 //Add additional code

```

List.clientScriptFileDialog

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The File Cabinet ID of client script file to be used in this list.
Type	number
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2016.1

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You attempted to set this value when the <code>List.clientScriptModulePath</code> property value has already been specified. For more information, see List.clientScriptModulePath .

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 list.clientScriptModulePath = 123;
7 ...
8 //Add additional code

```

List.clientScriptModulePath



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The relative path to the client script file to be used in this list.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2016.2

Errors

Error Code	Thrown If
PROPERTY_VALUE_CONFLICT	You attempted to set this value when the List.clientScriptModulePath property value has already been specified. For more information, see List.clientScriptModulePath .

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 objList.clientScriptModulePath = 'SuiteScripts/listBehavior.js';
4 ...
5 //Add additional code

```

List.style



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The display style for this list. For more information about possible values, see serverWidget.ListStyle .
Type	string

Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 list.style = serverWidget.ListStyle.REPORT;
7 ...
8 //Add additional code

```

List.title

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The list title.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 var title = list.title;
7 ...
8 //Add additional code

```

serverWidget.ListColumn

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A list column
---------------------------	---------------

	For a complete list of this object's methods and properties, see ListColumn Object Members .
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 var listcolumn = list.addColumn({
7   id : 'column1',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text',
10  align : serverWidget.LayoutJustification.RIGHT
11 });
12 ...
13 //Add additional code

```

ListColumn.addParamToURL(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a URL parameter (optionally defined per row) to the list column's URL.
Returns	serverWidget.ListColumn object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.param	string	required	The name for the parameter.
options.value	string	required	The value for the parameter.
options.dynamic	boolean	optional	If true, then the parameter value is actually an alias that is calculated per row.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 var listcolumn = list.addColumn({
7   id : 'column1',
8   type : serverWidget.FieldType.URL,
9   label : 'URL',
10 });
11 listcolumn.addParamToURL({
12   param : 'index',
13   value : '3'
14 });
15 ...
16 //Add additional code

```

ListColumn.setURL(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the base URL for the list column.
Returns	serverWidget.ListColumn
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.1

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.url	string	required	The base URL or a column in the data source that returns the base URL for each row
options.dynamic	boolean	optional	If true, then the URL is actually an alias that is calculated per row.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({

```

```

4     title : 'Simple List'
5 });
6 var listcolumn = list.addColumn({
7     id : 'column1',
8     type : serverWidget.FieldType.URL,
9     label : 'URL',
10 });
11 listcolumn.setURL({
12     url : 'http://www.netsuite.com'
13 })
14 ...
15 //Add additional code

```

ListColumn.label

i Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	This list column label.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2016.1

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4     title : 'Simple List'
5 });
6 var listcolumn = list.addColumn({
7     id : 'column1',
8     type : serverWidget.FieldType.URL,
9     label : 'URL',
10 });
11 var label = listcolumn.label;
12 ...
13 //Add additional code

```

serverWidget.Sublist

i Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	A sublist on a serverWidget.Form or an serverWidget.Assistant object. To add a sublist, use Assistant.addSublist(options) or Form.addSublist(options) .
	i Note: This object is read-only except for instances created via the serverWidget module using Suitelets or beforeLoad user event scripts.

For a complete list of this object's methods and properties, see [Sublist Object Members](#).

Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 ...
12 //Add additional code

```

Sublist.addButton(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a button to a sublist.
Returns	serverWidget.Button
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The internal ID name of the button. The internal ID must be in lowercase and without spaces.
options.label	string	required	The label for the button.
options.functionName	string	optional	The function name to be triggered on a button click.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 sublist.addButton({
12   id : 'buttonid',
13   label : 'Test'
14 });
15 ...
16 //Add additional code

```

Sublist.addField(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a field to a sublist.
Returns	serverWidget.Field object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	The internal ID for this field. The internal ID must be in lowercase and without spaces.	2015.2
options.label	string	required	The UI label for this field.	2015.2
options.type	string	required	The field type. Use the serverWidget.FieldType enum to set this value. The DATETIME, FILE, HELP, INLINEHTML, LABEL, LONGTEXT, and RICHTEXT values are not supported with this method. The MULTISELECT value is not supported for SuiteScript 2.0 Suitelets.	2015.2

Parameter	Type	Required / Optional	Description	Since
			<p>Note: If you have set the type parameter to SELECT, and you want to add custom options to the select field, you must set source to NULL.</p>	
options.source	string	optional	<p>The internalId or scriptId of the source list for this field.</p> <p>Use this parameter if you are adding a select (List/Record) type of field.</p> <p>Note: If you want to add custom options on a select field, you must set the source parameter to NULL.</p> <p>Important: After you create a select or multi-select field that is sourced from a record or list, you cannot add additional values with <code>Field.addSelectOption(options)</code>. The select values are determined by the source record or list.</p>	2015.2
options.container	string	optional	<p>The internal ID of the tab or field group to add the field to.</p> <p>Use this parameter to specify either a tab or a field group to place the field in.</p> <p>By default, the field is added to the main section of the form.</p>	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 /**
3 * @NApiVersion 2.0
4 * @NScriptType suitelet
5 */
6 define(['N/ui/serverWidget'], function(serverWidget) {
7     return Object.freeze({
8         onRequest: function(context) {
9             var form = serverWidget.createForm({
10                 title: 'Simple Form'
11             });
12             var sublist = form.addSublist({
13                 id: 'sublist',
14                 type: serverWidget.SublistType.INLINEEDITOR,
15                 label: 'Inline Editor Sublist'
16             });
17             sublist.addField({
18                 id: 'fieldid',
19                 type: serverWidget.FieldType.DATE,
20                 label: 'Date'
21             });
22             context.response.writePage(form);

```

```

23     }
24   });
25 });
26 //Add additional code

```

Sublist.addMarkAllButtons()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a Mark All and an Unmark All button to a LIST type of sublist. See the help topic Using Buttons in NetSuite for information on how Mark All/Unmark All Buttons works in NetSuite.
Returns	A serverWidget.Button[] object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.LIST,
9   label : 'List Sublist'
10});
11 sublist.addMarkAllButtons();
12 ...
13 //Add additional code

```

Sublist.addButton()

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a Refresh button to a LIST type of sublist.
Returns	serverWidget.Button object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module

Since	2015.2
-------	--------

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.LIST,
9   label : 'List Sublist'
10 });
11 sublist.addRefreshButton();
12 ...
13 //Add additional code

```

Sublist.getField(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a Field object on a sublist.
Returns	serverWidget.Field
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2016.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The field internal ID (for example, use item as the ID for the Item field). For more information about supported sublists, internal IDs, and field IDs, see the SuiteScript Records Browser .

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```
1 //Add additional code
```

```

2 ...
3 var itemField = form.getSublist({id: 'item'}).getField({id: 'item'});
4 ...
5 //Add additional code

```

Sublist.getSublistValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Gets a field value on a sublist.
Returns	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The internal ID of a field.
options.line	number	required	The line number for this field.
 Note: The first line number on a sublist is 0 (not 1).			

Errors

Error Code	Thrown If
SSS_MISSING_REQD_ARGUMENT	A required parameter is not passed.
YOU_CANNOT_CALL_1_METHOD_ON_SUBRECORD_FIELD_SUBLIST_2_FIELD_3	You called {1} method on subrecord field. Sublist: {2}, field: {3}.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var sublistvalue = sublist.getSublistValue({
4   id : 'quantity',
5   line: 1

```

```

6  })
7 ...
8 //Add additional code

```

Sublist.setSublistValue(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets the value of a sublist field.
Returns	void
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The internal ID name of the line item field being set.
options.line	number	required	<p>The line number for this field.</p> <p> Note: The first line number on a sublist is 0 (not 1).</p>
options.value	string	required	<p>The value for the field being set.</p> <p> Note: Checkbox fields accept string ('T'/'F') values, not boolean.</p>

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title: 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id: 'sublist',
8   type: serverWidget.SublistType.INLINEEDITOR,
9   label: 'Inline Editor Sublist'
10});
11

```

```

12 sublist.addField({
13   id: 'sublist',
14   type: ui.FieldType.TEXT,
15   label: 'Text Field'
16 });
17
18 sublist.setSublistValue({
19   id: 'sublist',
20   line: 2,
21   value: "Text"
22 });
23 ...
24 //Add additional code

```

Sublist.updateTotalingFieldId(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Sets a field as a totalling column, which is used to calculate and display a running total for the sublist.
Returns	serverWidget.Sublist object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.id	string	required	The internal ID of the field used to calculate the total field.

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

 **Note:** This script sample uses the define function, which is required for an entry point script (a script you attach to a script record and deploy). You must use the require function if you want to copy the script into the SuiteScript Debugger and test it. For more information, see [SuiteScript 2.0 Global Objects](#).

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4  */
5 define(['N/ui/serverWidget', 'N/record'], function(serverWidget, record){
6   return {

```

```

7 |     onRequest: function(params) {
8 |         var form = serverWidget.createForm({
9 |             title: 'Simple Form'
10 |         });
11 |         var sublistObj2 = form.addSublist({
12 |             id: 'mylist',
13 |             type: serverWidget.SublistType.INLINEEDITOR,
14 |             label: 'List'
15 |         });
16 |         sublistObj2.addField({
17 |             id: 'description',
18 |             type: serverWidget.FieldType.TEXT,
19 |             label: 'Description'
20 |         });
21 |         sublistObj2.addField({
22 |             id: 'amount',
23 |             type: serverWidget.FieldType.CURRENCY,
24 |             label: 'Amount'
25 |         });
26 |         sublistObj2.updateTotallingFieldId({
27 |             id: 'amount'
28 |         });
29 |         sublistObj2.setSublistValue({
30 |             id: 'description',
31 |             line: 0,
32 |             value: 'foo'
33 |         });
34 |         sublistObj2.setSublistValue({
35 |             id: 'amount',
36 |             line: 0,
37 |             value: '10'
38 |         });
39 |         sublistObj2.setSublistValue({
40 |             id: 'description',
41 |             line: 1,
42 |             value: 'bar'
43 |         });
44 |         sublistObj2.setSublistValue({
45 |             id: 'amount',
46 |             line: 1,
47 |             value: '15'
48 |         });
49 |         form.addSublist({
50 |             id: 'dummy',
51 |             type: serverWidget.SublistType.STATICLIST,
52 |             label: 'Dummy'
53 |         });
54 |         params.response.writePage(form);
55 |     }
56 | };
57 | });

```

Sublist.updateUniqueIdFieldId(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Indicates a field that must have unique values across the rows in the sublist.
	 Note: This method is available on inlineeditor and editor sublists only.
Returns	serverWidget.Sublist object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None

Module	N/ui/serverWidget Module
Since	2015.2

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description
options.id	string	required	The internal ID of the field that you want to contain unique values

Syntax

The following code sample uses the Sublist.updateUniqueId(options) method to set the **Date** field as unique. This means that a user will not be able to enter two rows with the same date on the sublist.

 Important:	The following code sample shows the syntax for this member. It is not a functional example. For a complete script example, see N/ui/serverWidget Module Script Samples .
---	--

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 sublist.addField({
12   id : 'fieldid',
13   type : serverWidget.FieldType.DATE,
14   label : 'Date'
15 });
16 sublist.updateUniqueId({
17   id : 'fieldid'
18 })
19 ...
20 //Add additional code

```

Sublist.displayType

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Property Description	The display style for a sublist. Use the serverWidget.SublistDisplayType enum to set this value.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 sublist.displayType = serverWidget.SublistDisplayType.HIDDEN;
12 ...
13 //Add additional code

```

Sublist.helpText

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The inline help text for a sublist.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 sublist.helpText = "Help Text Goes Here.";
12 ...
13 //Add additional code

```

Sublist.label

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The label for this sublist.
-----------------------------	-----------------------------

Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 var label = sublist.label;
12 ...
13 //Add additional code

```

Sublist.lineCount

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The number of line items on a sublist.
	Note: The first line number on a sublist is 0 (not 1).
Type	number (read-only)
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({

```

```

7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 var numLines = sublist.lineCount;
12 ...
13 //Add additional code

```

serverWidget.Tab

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	<p>A tab or subtab on a serverWidget.Form object.</p> <p>You can add a new tab or subtab to a form using one of the following methods:</p> <ul style="list-style-type: none"> ▪ Form.addSubtab(options) ▪ Form.addTab(options) ▪ Form.insertSubtab(options) ▪ Form.insertTab(options) <p>The internal ID must be in lowercase, contain no spaces, and include the prefix custpage if you are adding the field to an existing page.</p> <p>Note: In order for your tab to appear on your form, there must be at least one object assigned to the tab. Otherwise, the tab will not appear.</p> <p>Note: If you have less than two tabs on your form, the tab will not appear. Instead the fields assigned to the tab will appear at the bottom of the form.</p> <p>For a complete list of this object's properties, see Tab Object Members.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var tab = form.addTab({
7   id : 'tabid1',
8   label : 'Tab 1'
9 });
10 var tab = form.addTab({
11   id : 'tabid2',
12   label : 'Tab 2'
13 });
14
15

```

```

16 | form.addField({
17 |   id : 'custpage_tabid1',
18 |   type: ui.FieldType.TEXT,
19 |   label: 'Tab 1 Field'
20 | });
21 |
22 | form.addField({
23 |   id : 'custpage_tabid2',
24 |   type: ui.FieldType.TEXT,
25 |   label: 'Tab 2 Field'
26 | });
27 | //Add additional code

```

Tab.helpText



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The inline help text for a tab or subtab.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var tab = form.addTab({
7   id : 'tabid',
8   label : 'Tab'
9 });
10 tab.helpText = 'Help Text Goes Here';
11 ...
12 //Add additional code

```

Tab.label



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The label for a tab or subtab.
Type	string
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module

Since	2015.2
--------------	--------

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var tab = form.addTab({
7   id : 'tabid',
8   label : 'Tab'
9 });
10 var label = tab.label;
11 ...
12 //Add additional code

```

serverWidget.createAssistant(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates an assistant object.
	i Note: The assistant cannot be used on externally available Suitelets.
Returns	serverWidget.Assistant object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.title	string	required	The title of the assistant. This title appears at the top of all assistant pages.	2015.2
options.hideNavBar	boolean true false	optional	Indicates whether to hide the navigation bar menu. By default, set to false. The header appears in the top-right corner on the assistant.	2015.2

Parameter	Type	Required / Optional	Description	Since
			If set to true, the header on the assistant is hidden from view.	

Syntax

⚠ Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4   title : 'Simple Assistant'
5 });
6 ...
7 //Add additional code

```

For more information, see the help topic [Sample Custom Assistant Script](#).

serverWidget.createForm(options)

i Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a form object.
Returns	<code>serverWidget.Form</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters

i Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.title</code>	string	required	The title of the form.	2015.2
<code>options.hideNavBar</code>	boolean true false	optional	<p>Indicates whether to hide the navigation bar menu.</p> <p>By default, set to false. The header appears in the top-right corner on the form.</p> <p>If set to true, the header on the assistant is hidden from view.</p>	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code ...
2 var form = serverWidget.createForm({
3   title : 'Simple Form'
4 });
5 ...
6 //Add additional code

```

serverWidget.createList(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a standalone list.
Returns	<code>serverWidget.List</code> object
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Governance	None
Module	N/ui/serverWidget Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
<code>options.title</code>	string	required	The title of the list.	2016.1
<code>options.hideNavBar</code>	boolean true false	optional	Indicates whether to hide the navigation bar menu. By default, set to false. The header appears in the top-right corner on the form. If set to true, the header on the assistant is hidden from view.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({

```

```

4     title : 'Simple List'
5 });
6 ...
7 //Add additional code

```

serverWidget.AssistantSubmitAction

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for submit actions performed by the user. This enum is used to set the value of the Assistant.getLastAction() . After a finish action is submitted, by default, the text “Congratulations! You have completed the <assistant title>” appears on the finish page. In a non-sequential process (steps are unordered), jump is used to move to the user’s last action.
	Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Values

Value
BACK
CANCEL
FINISH
JUMP
NEXT

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var assistant = serverWidget.createAssistant({
4     title : 'Simple Assistant'
5 });
6 ...
7 if (assistant.getLastAction() === serverWidget.AssistantSubmitAction.CANCEL) {
8     ...
9 }

```

```

10 ...
11 //Add additional code

```

serverWidget.FieldBreakType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported field break types. This enum is used to set the value of the breakType parameter when Field.updateBreakType(options) is called.
	Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Values

Value	Description
NONE	This is the default value for field break type.
STARTCOL	This value moves the field into a new column. Additionally, it disables automatic field balancing if set on any field.
STARTROW	This value places a field located outside of a field group on a new row. This value only works on fields with a Field Layout Type set to OUTSIDE, OUTSIDEABOVE or OUTSIDE BELOW. For more information, see serverWidget.FieldLayoutType and Field.updateLayoutType(options) .

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_text',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10});
11
12 field.updateLayoutType({
13   layoutType: serverWidget.FieldLayoutType.OUTSIDE
14 });
15
16 field.updateBreakType({
17   breakType : serverWidget.FieldBreakType.STARTROW

```

```

18 });
19 ...
20 //Add additional code

```

serverWidget.FieldDisplayType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported field display types. This enum is used to set the value of the displayType parameter when Field.updateDisplayType(options) is called.
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Values

Value	Description of Field Type
DISABLED	Prevents a user from changing the field
ENTRY	The sublist field appears as a data entry input field (for a select field without a checkbox)
HIDDEN	The field on the form is hidden.
INLINE	The field appears as inline text
NORMAL	The field appears as a normal input field (for non-sublist fields)
READONLY	The field is disabled but it is still selectable and scrollable (for textarea fields)

Syntax



Important: The following code snippets show the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_text',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 field.updateDisplayType({
12   displayType: serverWidget.FieldDisplayType.HIDDEN
13 });
14 ...

```

15 //Add additional code

serverWidget.FieldLayoutType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for the supported types of field layouts. This enum is used to set the value of the layoutType parameter when Field.updateLayoutType(options) is called.
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Values

Value	Description
STARTROW	This value makes the field appear first in a horizontally aligned field group in the normal field layout.
MIDROW	This value makes the field appear in the middle of a horizontally aligned field group in the normal field layout.
ENDROW	This value makes the field appear last in a horizontally aligned field group in the normal field layout.
OUTSIDE	This value makes the field appear outside (above or below based on form default) the normal field layout area.
OUTSIDE BELOW	This value makes the field appear below the normal field layout area. Using this enables you to position a field below a field group.
OUTSIDE ABOVE	This value makes the field appear above the normal field layout area. Using this enables you to position a field above a field group.
NORMAL	This value makes the fields appear in its default position.

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title: 'Simple Form'
5 });
6 var field = form.addField({
```

```

7   id: 'custpage_text',
8   type: serverWidget.FieldType.TEXT,
9   label: 'Text'
10 });
11 field.updateLayoutType({
12   layoutType: serverWidget.FieldLayoutType.OUTSIDEBELOW
13 });
14 ...
15 //Add additional code

```

serverWidget.FieldType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the values for supported field types. This enum is used to set the value of the type parameter when Form.addField(options) is called.
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p> <p>Important: Long text fields created with SuiteScript have a character limit of 100,000. Long text fields created with SuiteBuilder have a character limit of 1,000,000.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Values

CHECKBOX	LONGTEXT
CURRENCY	MULTISELECT
DATE	PASSWORD
DATETIME	PERCENT
DATETIMETZ	PHONE
EMAIL	SELECT
FILE	RADIO
FLOAT	RICHTEXT
HELP	TEXT
INLINEHTML	TEXTAREA
INTEGER	TIMEOFDAY
IMAGE	URL

LABEL	
-------	--

Consider the following as you work with these field types:

- The DATETIME field type is not supported with addField methods, you must specify DATETIMETZ.
- The FILE field type is available only for Suitelets and will appear on the main tab of the Suitelet page. FILE fields cannot be added to tabs, subtabs, sublists, or field groups and are not allowed on existing pages.
- The DATETIME, HELP, INLINEHTML, LABEL, LONGTEXT, and RICHTEXT field types are not supported with [Sublist.addField\(options\)](#).
- The INLINEHTML field type should be considered as a 'write-only' type of field just used to add a field on a form.
- The IMAGE field type is available only for fields that appear on list/staticlist sublists. You cannot specify an IMAGE field on a form.
- The MULTISELECT field type is not supported by SuiteScript 2.0 Suitelets.
- Radio buttons that are inside one container are exclusive. The method addField on form has an optional parameter container. For an example, see [FieldGroup.label](#).

For a description of these field types, and additional information regarding character limits and format restrictions, see the help topic [Table of Custom Field Type Descriptions](#).

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var field = form.addField({
7   id : 'custpage_text',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text'
10 });
11 ...
12 //Add additional code

```

serverWidget.FormPageLinkType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	<p>Holds the string values for supported page link types on a form. This enum is used to set the value of the type parameter when Form.addPageLink(options) is called.</p> <p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))

Module	N/ui/serverWidget Module
Since	2015.2

Values

Value	Description
BREADCRUMB	Link appears on the top-left corner after the system bread crumbs
CROSSLINK	Link appears on the top-right corner

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 form.addPageLink({
7   type : serverWidget.FormPageLinkType.CROSSLINK,
8   title : 'NetSuite',
9   url : 'http://www.netsuite.com'
10 })
11 ...
12 //Add additional code

```

serverWidget.LayoutJustification

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported justification layouts. This enum is used to set the value of the align parameter when List.addColumn(options) is called.
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Values

Value
CENTER

Value
LEFT
RIGHT

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 list.addColumn({
7   id : 'column1',
8   type : serverWidget.FieldType.TEXT,
9   label : 'Text',
10  align : serverWidget.LayoutJustification.RIGHT
11 });
12 ...
13 //Add additional code

```

serverWidget.ListStyle



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported list styles. This enum is used to set the value of the List.style property.
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Values

Value
GRID
REPORT
PLAIN
NORMAL

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var list = serverWidget.createList({
4   title : 'Simple List'
5 });
6 list.style = serverWidget.ListStyle.REPORT;
7 ...
8 //Add additional code

```

serverWidget.SublistDisplayType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for supported sublist display types. This enum is used to set the value of the Sublist.displayType property.
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (<code>(beforeLoad(scriptContext)</code>)
Module	N/ui/serverWidget Module
Since	2015.2

Values

Value
HIDDEN
NORMAL

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });

```

```

11 sublist.displayType = serverWidget.SublistDisplayType.HIDDEN;
12 ...
13 //Add additional code

```

serverWidget.SublistType



Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Holds the string values for valid sublist types. This enum is used to define the type parameter when Form.addSublist(options) is called
	<p>Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation utilizes the term enumeration (or enum) to describe the following: a plain JavaScript object with a flat, map-like structure. Within this object, each key points to a read-only string value.</p>
Supported Script Types	SuiteScript 2.0 Suitelet Script Type and SuiteScript 2.0 User Event Script Type (beforeLoad(scriptContext))
Module	N/ui/serverWidget Module
Since	2015.2

Values

Value	Description
INLINEEDITOR	These types of sublists are both fully editable. The only difference between these types is their appearance in the UI:
EDITOR	<ul style="list-style-type: none"> ■ With an inline editor sublist, a new line is displayed at the bottom of the list after existing lines. To add a line, a user working in the UI clicks inside the new line and adds a value to each column as appropriate. Examples of this style include the Item sublist on the sales order record and the Expense sublist on the expense report record. ■ With an editor sublist, a user in the UI adds a new line by working with fields that are displayed above the existing sublist lines. This style is not common on standard NetSuite record types.
LIST	This type of sublist has a fixed number of lines. You can update an existing line, but you cannot add lines to it.
STATICLIST	<p>Note: To make a field within a LIST type sublist editable, use Field.updateDisplayType(options) and the enum serverWidget.FieldDisplayType to update the field display type to ENTRY.</p>

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/ui/serverWidget Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form with Inline Editor type Sublist'

```

```

5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.INLINEEDITOR,
9   label : 'Inline Editor Sublist'
10 });
11 ...
12 //Add additional code

```

The following code snippet shows how to make a field within a LIST type sublist editable by updating the fieldDisplayType to ENTRY.

```

1 //Add additional code
2 ...
3 var form = serverWidget.createForm({
4   title : 'Simple Form with List type Sublist'
5 });
6 var sublist = form.addSublist({
7   id : 'sublist',
8   type : serverWidget.SublistType.LIST,
9   label : 'List Type Sublist'
10 });
11 var internalId = sublist.addField({
12   id : 'id',
13   label : 'Internal ID',
14   type : serverWidget.FieldType.TEXT
15 });
16 internalId.updateDisplayType({displayType: serverWidget.FieldDisplayType.ENTRY});
17 ...
18 //Add additional code

```

N/url Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Use the N/url module to determine URL navigation paths within NetSuite and format URL strings.

- [N/url Module Members](#)
- [N/url Module Script Samples](#)



Note: If you have any hard-coded references to external URLs for Suitelets with the forms.netsuite.com domain, you must update these references. As of 2020.1, any external URL references with the old format will result in broken links. For access or redirection from another script to a Suitelet, the best practice is to use [url.resolveScript\(options\)](#) to discover the URL instead of hard-coding the URL. For more information about NetSuite domains, see the help topic [Understanding NetSuite URLs](#).

N/url Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	url.format(options)	string	Client and server scripts	Converts (serializes) URL query parameters into a string.
	url.resolveDomain(options)	string	Client and server scripts	Returns a domain name for a NetSuite account.
	url.resolveRecord(options)	string	Client and server scripts	Returns an internal URL string to a NetSuite record.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	url.resolveScript(options)	string	Server scripts	Returns an external or internal URL string to a script.
	url.resolveTaskLink(options)	string	Client and server scripts	Returns an internal URL for a tasklink.
Enum	url.HostType	enum	Client and server scripts	An enum used to populate the hostType parameter of the url.resolveDomain(options) method.

N/url Module Script Samples

The following script samples show how to use the N/url module.

 **Note:** This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to retrieve the relative URL of a record. With the internal ID value used in this sample, the returned output would be /app/accounting/transactions/salesord.nl?id=6&e=T&compid=' , followed by the NetSuite account ID.

 **Important:** The value used in this sample for the recordId field is a placeholder. Before using this sample, replace the recordId field value with a valid value from your NetSuite account. If you run a script with an invalid value, an error may occur.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/url'],function(url) {
6   var output = url.resolveRecord({
7     recordType: 'salesorder',
8     recordId: 6,
9     isEditMode: true
10   });
11 });

```

 **Note:** This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to generate an absolute URL to a specific resource.

 **Important:** The value used in this sample for the recordId field is a placeholder. Before using this sample, replace the recordId field value with a valid value from your NetSuite account. If you run a script with an invalid value, an error may occur.

```

1 /**
2  * @NApiVersion 2.x

```

```

3  */
4
5 require(['N/url', 'N/record'], function(url, record) {
6   function resolveRecordUrl() {
7     var scheme = 'https://';
8     var host = url.resolveDomain({
9       hostType: url.HostType.APPLICATION
10    });
11    var relativePath = url.resolveRecord({
12      recordType: record.Type.SALES_ORDER,
13      recordId: 6,
14      isEditMode: true
15    });
16    var output = scheme + host + relativePath;
17  }
18  resolveRecordUrl();
19 });

```

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to get the domain for calling a RESTlet.

Important: The value used in this sample for the `accountId` field is a placeholder. Before using this sample, replace the `accountId` field value with a valid value from your NetSuite account. If you run a script with an invalid value, an error may occur.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/url'], function(url) {
6   function resolveDomainUrl() {
7     var sCompId = 'MSTRWLF';
8     var output = url.resolveDomain({
9       hostType: url.HostType.RESTLET,
10      accountId: sCompId
11    });
12  }
13  resolveDomainUrl();
14 });

```

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample shows how to create a URL and do send a secure HTTPS POST request to that URL with an empty body. The server's response is logged.

Important: The value used in this sample for the `scriptId` and `deploymentId` fields are placeholders. Before using this sample, replace the `scriptId` and `deploymentId` values with valid values from your NetSuite account. If you run a script with an invalid value, an error may occur.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/url', 'N/https'], function(url, https) {
6   var script = 'customscript1';

```

```

7  var deployment = 'customdeploy1';
8  var parameters = "";
9  try {
10    var suiteletURL = url.resolveScript({
11      scriptId:script,
12      deploymentId: deployment
13    });
14    var response = https.post({
15      url: suiteletURL,
16      body: parameters
17    });
18    log.debug(response.body.toString());
19  }
20  catch(e) {
21    log.error(e.toString());
22  }
23 });

```

url.format(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a serialized representation of an object containing query parameters. Use the returned value to build a URL query string.
Returns	URL as a string
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/url Module
Since	2015.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.domain	string	required	The domain name.	2015.1
options.params	Object	required	Additional URL parameters as name/value pairs.	2015.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/url Module Script Samples](#).

For a script that uses the following code snippet, the returned output is `http://fruitland.com?fruit=grape&seedless=true&variety=Concord+Giant&PLU=4272`, expressed as a string.

```

1 //Add additional code
2 ...

```

```

3 var output = url.format({
4   domain: 'http://fruitland.com',
5   params: {
6     fruit: 'grape',
7     seedless: true,
8     variety: 'Concord Giant',
9     PLU: 4272
10   }
11 });
12 ...
13 //Add additional code

```

url.resolveDomain(options)

Method Description	Returns a domain name for a NetSuite account.
Returns	string
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/url Module
Since	2017.1

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.hostType	string	required	The type of domain name you want to retrieve. Set this value using the <code>url.HostType</code> enum.	2017.1
options.accountId	string	optional	The NetSuite account ID for which you want to retrieve data. If no account is specified, the system returns data on the account that is running the script. You can find the account ID at Setup > Company > Company Information in the Account ID field.	2017.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/url Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var output = url.resolveDomain({
4   hostType: url.HostType.APPLICATION,
5   accountId: '012345'
6 });
7 ...
8 //Add additional code

```

url.resolveRecord(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the URL string to a NetSuite record.
Returns	URL to a NetSuite record as a string
Supported Script Types	Client and server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/url Module
Since	2015.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.isEditMode	boolean true false	required	If set to true, returns a URL for the record in Edit mode. If set to false, returns a URL for the record in View mode The default value is View.	2015.1
options.recordId	string	required	The internal ID of the target record instance.	2015.1
options.recordType	string	required	The type of record. For example, 'transaction'.	2015.1
options.params	Object	optional	Object used to add parameters for a custom URL. For example, a query to a database or to a search engine	2015.1

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/url Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var output = url.resolveRecord({
4   recordType: 'salesorder',
5   recordId: 6,
6   isEditMode: true
7 });
8 ...
9 //Add additional code

```

url.resolveScript(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an external or internal URL string to a script.
Returns	The URL as a string
Supported Script Types	All server scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/url Module
Since	2015.1

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.deploymentId	string	required	The internal ID of the deployment script	2015.1
options.scriptId	string	required	The internal ID of the script. The ID must identify a RESTlet or a Suitelet.	2015.1
options.params	Object	optional	The object containing name/value pairs to describe the query.	2015.1
options.returnExternalUrl	boolean	optional	Indicates whether to return the external URL. By default, the internal URL is returned (i.e., the default value is false).	2015.1

Note: Currently, it is not possible to call a Suitelet using its internal URL from a server script. You can either set the options.returnExternalURL parameter to true or you need to pass the cookie from the server (and concatenate "https://" appropriately). Also note that a cookie cannot be retrieved in afterSubmit entry point; although the request parameter from which it could be retrieved is available in beforeLoad and it could be transferred to afterSubmit. Calling a Suitelet using its internal URL is only supported in client scripts because the browser automatically adds the cookie header to the request.

Errors

Error Code	Message	Thrown If
SSS_INVALID_URL		You attempt to call a Suitelet using its internal URL from a server script.

Error Code	Message	Thrown If
SSS_INVALID_URL_CATEGORY		You used this method in a client script. This method is only supported in server scripts.

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/url Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var output = url.resolveScript(
4    scriptId: 'custom_script',
5     deploymentId: 'custom_script_deployment',
6     returnExternalUrl: true
7 );
8 ...
9 //Add additional code

```

url.resolveTaskLink(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the internal URL to a NetSuite tasklink.
Returns	The URL as a string
Supported Script Types	All server and client scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/url Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.id	string	required	Internal ID for the tasklink. Note: Each page in NetSuite has a unique tasklink Id associated with it for a specific record type. You can determine the tasklink for a page within NetSuite by viewing the HTML page source. Search for a string similar to the following, where LIST_SCRIPT refers to the TASKLINK: onclick="nIIPopupHelp('LIST_SCRIPT','help').	2015.1
options.params	Map	optional	The Map object containing name/value pairs to describe the query.	2015.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/url Module Script Samples](#).

```

1 //Add additional code
2 ...
3 u = url.resolveTaskLink('SRCH_JOB', p);
4 ...
5 //Add additional code

```

url.HostType



Note: The content in this help topic pertains to SuiteScript 2.0.



Note: JavaScript does not include an enumeration type. The SuiteScript 2.0 documentation uses the term enumeration (or enum) to describe a plain JavaScript object with a flat, map-like structure. In this object, each key points to a read-only string value.

Enum Description	Enumeration whose string values each describe a category of domain name. This enum is used to set the value of the hostType parameter of the url.resolveDomain(options) method.
Type	enum
Supported Script Types	All server and client scripts For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/url Module
Since	2017.1

Values

Value	Description	Sample Result
APPLICATION	The domain for UI access.	<accountID>.app.netsuite.com <accountID> is replaced with your NetSuite account number.
FORM	The domain for forms hosted online, usually in Suitelets.	<accountID>.extforms.netsuite.com <accountID> is replaced with your NetSuite account number.
RESTLET	The domain for calling a RESTlet from an external source.	<accountID>.restlets.api.netsuite.com <accountID> is replaced with your NetSuite account number.
SUITETALK	The domain for SOAP web services requests.	<accountID>.suitetalk.api.netsuite.com <accountID> is replaced with your NetSuite account number.

For more information about NetSuite domains, see the help topic [Understanding NetSuite URLs](#).



Warning: The results returned, as shown in the sample results column, may change without notice. Because these values can change, your scripts must dynamically discover domain names. For more details, see the help topic [NetSuite Accounts Are Hosted in the Cloud](#).

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/url Module Script Samples](#).

```

1 //Add additional code
2 ...
3 var output = url.resolveDomain({
4     hostType: url.HostType.APPLICATION,
5     accountId: '012345'
6 });
7 ...
8 //Add additional code

```

N/util Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/util module when you want to access methods that verify object and primitive types in a SuiteScript 2.0 script.

Each method (for example, `util.isArray(obj)`) returns a boolean value, based on evaluation of the `obj` parameter.

If you need to identify a type specific to SuiteScript 2.0, use the `toString()` global method.



Note: The util Object can be accessed globally or by loading this module. Load the N/util module when you want to manually access the util module members, such as for testing purposes. For more information about global objects, see [SuiteScript 2.0 Global Objects](#).

- [N/util Module Members](#)
- [N/util Module Script Samples](#)

N/util Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	util.isArray(obj)	boolean	Client and server-side scripts	Returns true if the <code>obj</code> parameter is a JavaScript Array object and false otherwise.
	util.isBoolean(obj)	boolean	Client and server-side scripts	Returns true if the <code>obj</code> parameter is a JavaScript Boolean and false otherwise.
	util.isDate(obj)	boolean	Client and server-side scripts	Returns true if the <code>obj</code> parameter is a JavaScript Date object and false otherwise.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	util.isAsyncFunction(obj)	boolean	Client and server-side scripts	Returns true if the obj parameter is JavaScript Async Function and false otherwise.
	utilisFunction(obj)	boolean	Client and server-side scripts	Returns true if the obj parameter is a JavaScript Function or Async Function and false otherwise.
	util.isNumber(obj)	boolean	Client and server-side scripts	Returns true if the obj parameter is a JavaScript Number object or a value that evaluates to a Number object, and false otherwise.
	utilisObject(obj)	boolean	Client and server-side scripts	Returns true if the obj parameter is a plain JavaScript object (new Object() or {} for example), and false otherwise.
	util.isRegExp(obj)	boolean	Client and server-side scripts	Returns true if the obj parameter is a JavaScript RegExp object or a value that evaluates to a RegExp object, and false otherwise.
	util.isString(obj)	boolean	Client and server-side scripts	Returns true if the obj parameter is a JavaScript String object or a value that evaluates to a String object, and false otherwise.
	util.each(iterable, callback)	Object or Array	Client and server-side scripts	Iterates over each member in an Object or Array.
	util.extend(receiver, contributor)	Object	Client and server-side scripts	Copies the properties in a source object to a destination object.

N/util Module Script Samples

The following script samples demonstrate how to use the features of the N/util module.

Sample 1: Iterate through an object

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample creates a sales order record. It uses the `util.each(iterable, callback)` method to set record fields based on the values in an iterable object. Make sure to replace hard-coded values (such as record IDs) with valid values from your NetSuite account.

```

1 /**
2  * @NApiVersion 2.x
3 */
4
5 require(['N/record'], function(record){
6     // Create a sales order
7     var rec = record.create({}

```

```

8     type: 'salesorder',
9     isDynamic: true
10    });
11    rec.setValue({
12      fieldId: 'entity',
13      value: 107
14    });
15
16 // Set up an object containing an item's internal ID and the corresponding quantity
17 var itemList = {
18   39: 5,
19   38: 1
20 }
21
22 // Iterate through the object and set the key-value pairs on the record
23 util.each(itemList, function(quantity, itemId){ // (5, 39) and (1, 38)
24   rec.selectNewLine('item');
25   rec.setCurrentSublistValue('item', 'item', itemId);
26   rec.setCurrentSublistValue('item', 'quantity', quantity);
27   rec.commitLine('item');
28 });
29
30 var id = rec.save();
31 });

```

util.isArray(obj)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a JavaScript Array object and false otherwise.
Returns	boolean
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Global object	util Object
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2016.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see N/util Module Script Sample.

```

1 //Add additional code
2 ...
3 var records = ["Sales Order", "Invoice", "Item Fulfillment"];
4 util.isArray(records); // returns true

```

```

5
6 var record = "Sales Order";
7 util.isArray(record); // returns false
8 ...
9 //Add additional code

```

util.isBoolean(obj)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a JavaScript Boolean and false otherwise.
Returns	boolean
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see [N/util Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var flag = true;
4 util.isBoolean(flag); // returns true
5 util.Boolean(true); // returns true
6
7 util.Boolean(1); // returns false
8 ...
9 //Add additional code

```

util.isDate(obj)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a JavaScript Date object and false otherwise.
Returns	boolean
Supported Script Types	All script types

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see [N/util Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var todaysDate = new Date();
4 util.isDate(todaysDate);    //returns true
5 util.isDate(new Date());   //returns true
6
7 var today = "September 28, 2015";
8 util.isDate(today);        //returns false
9 ...
10 //Add additional code

```

util.isAsyncFunction(obj)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a JavaScript AsyncFunction and false otherwise.
Returns	boolean
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2020.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2020.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see N/util Module Script Sample.

```

1 //Add additional code
2 ...
3 function func1(){
4     var x = 1;
5 }
6 var func2 = function(){}
7     async function async1(){
8         var x = 2;
9     }
10 var async2 = async function(){}
11
12 util.isAsyncFunction(func1); //returns false
13 util.isAsyncFunction(func2); //returns false
14 util.isAsyncFunction(async1); //returns true
15 util.isAsyncFunction(async2); //returns true
16 ...
17 //Add additional code

```

util.isFunction(obj)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a JavaScript Function or AsyncFunction and false otherwise.
Returns	boolean
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see N/util Module Script Sample.

```

1 //Add additional code
2 ...
3 function test() {};
4 var test2 = function() {};
5

```

```

6 util.isFunction(test);    // returns true
7 util.isFunction(test2);   // returns true
8 ...
9 //Add additional code

```

util.isNumber(obj)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a JavaScript Number object or primitive, and false otherwise.
Returns	boolean
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see [N/util Module Script Sample](#).

```

1 //Add additional code
2 ...
3 util.isNumber(112);           // returns true
4 util.isNumber("112");         // returns false
5 util.isNumber(NaN);          // returns true
6
7 var testNum = 112;
8 util.isNumber(testNum.valueOf()); // returns true
9 ...
10 //Add additional code

```

utilisObject(obj)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a plain JavaScript object (<code>new Object()</code> or <code>{}</code> for example), and false otherwise. Use this method, for example, to verify that a variable is a JavaScript object and not a JavaScript Function.
---------------------------	---

Returns	boolean
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2016.1

Syntax

 **Important:** The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see [N/util Module Script Sample](#).

```

1 //Add additional code
2 ...
3 util.isObject({});           // returns true
4 util.isObject(function() {}); // returns false
5 ...
6 //Add additional code

```

util.isRegExp(obj)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a JavaScript RegExp object, and false otherwise.
Returns	boolean
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see N/util Module Script Sample.

```

1 //Add additional code
2 ...
3 util.isRegExp(/this is a regexp/);           // returns true
4 util.isRegExp(new RegExp('this is another regexp')); // returns true
5 ...
6 //Add additional code

```

util.isString(obj)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the obj parameter is a JavaScript String object or primitive, and false otherwise
Returns	boolean
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
obj	Object Primitive	required	Object for which you want to verify the type.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see N/util Module Script Sample.

```

1 //Add additional code
2 ...
3 util.isString('');                      // returns true
4 util.isString('a string');                // returns true
5
6 var myString = new String('another string');
7 util.isString(myString);                  // returns true
8
9 util.isString(null);                     // returns false
10 ...
11 //Add additional code

```

util.each(iterable, callback)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Iterates over each member in an Object or Array. This method calls the callback function on each member of the iterable.
Returns	The original collection as an Object Array
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Parameters

Parameter	Type	Required / Optional	Description	Since
iterable	Object Array	required	The data collection to iterate on.	2016.1
callback	Function	required	Takes the custom logic that you want to execute on each member of your collection of data.	2016.1

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a full script sample, see N/util Module Script Sample.

```

1 //Add additional code
2 ...
3 //Iterate through the object and set the key-value pairs on the record
4 util.each(itemList, function(quantity, itemId){
5     rec.selectNewLine('item');
6     rec.setCurrentSublistValue('item','item',itemId);
7     rec.setCurrentSublistValue('item','quantity',quantity);
8     rec.commitLine('item');
9 });
10 ...
11 //Add additional code

```

util.extend(receiver, contributor)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Copies the properties in a source object to a destination object. You can use this method to merge two objects.
---------------------------	--

Returns	The destination object.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/util Module
Since	2016.1

Syntax

Important: The following code snippets shows the syntax for this member. It is not a functional example. For a full script sample, see [N/util Module Script Sample](#).

This snippet shows combining two objects without the same keys:

```

1 //Add additional code
2 ...
3     var colors = {};
4     var firstSet = {'color1':'red',
5                     'color2':'yellow',
6                     'color3':'blue'};
7     var secondSet = {'color4':'green',
8                     'color5':'orange',
9                     'color6':'violet'
10    };
11
12 // Extends colors object with the information in firstSet
13 // Colors will get {'color1':'red','color2':'yellow','color3':'blue'}
14 util.extend(colors, firstSet);
15
16 // Extends colors object with the information in secondSet
17 // Colors will get {'color1':'red','color2':'yellow','color3':'blue','color4':'green','color5':'orange','color6':'violet'}
18 util.extend(colors, secondSet);
19 });
20 ...
21 //Add additional code

```

The following snippet shows overriding two objects with a few similar keys:

```

1 //Add additional code
2 ...
3     var colors = {};
4     var firstSet = {'color1':'red',
5                     'color2':'yellow',
6                     'color3':'blue'
7    };
8     var secondSet = {'color2':'green',
9                     'color3':'orange',
10                    'color4':'violet'
11    };
12
13 // Extends colors object with the information in firstSet
14 // Colors will get {'color1':'red','color2':'yellow','color3':'blue'}
15 util.extend(colors, firstSet);
16
17 // Extends colors object with the information in secondSet and overrides the value if there are similar keys
18 // Colors will get {'color1':'red','color2':'green','color3':'orange','color4':'violet'}
19 util.extend(colors, secondSet);
20
21     var x = 0;
22 });
23 ...

```

24 | //Add additional code

N/workflow Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the N/workflow module when you want to initiate new workflow instances or trigger existing workflow instances.

- [N/workflow Module Members](#)
- [N/workflow Module Script Sample](#)

N/workflow Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	workflow.initiate(options)	number	Server scripts	<p>Initiates a workflow on-demand. This method is the programmatic equivalent of the Initiate Workflow Action action in SuiteFlow.</p> <p>Returns the internal ID of the workflow instance used to track the workflow against the record.</p>
	workflow.trigger(options)	number	Server scripts	<p>Triggers a workflow on a record. The actions and transitions of the workflow are evaluated for the record in the workflow instance, based on the current state for the workflow instance.</p> <p>Returns the internal ID of the workflow instance used to track the workflow against the record.</p>

N/workflow Module Script Sample



Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample searches for a specific workflow deployed on the customer record and then executes it.



Important: This script sample uses placeholder values for the customer recordId and workflowId. Before using this sample, replace these IDs with valid values from your NetSuite account. If you run a script with an invalid value, the system may throw an error.

```
1 /**
2  * @NApiVersion 2.x
```

```

3  */
4
5 require(['N/workflow', 'N/search', 'N/error', 'N/record'],
6   function(workflow, search, error, record) {
7     function initiateWorkflow() {
8       var workflowInstanceId = workflow.initiate({
9         recordType: 'customer',
10        recordId: 24,
11        workflowId: 'customworkflow_myWorkFlow'
12      });
13      var customerRecord = record.load({
14        type: record.Type.CUSTOMER,
15        id: 24
16      });
17    }
18    initiateWorkflow();
19  });

```

workflow.initiate(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Initiates a workflow on-demand. This method is the programmatic equivalent of the Initiate Workflow Action action in SuiteFlow. Returns the internal ID of the workflow instance used to track the workflow against the record. To asynchronously initiate a workflow, see task.WorkflowTriggerTask .
Returns	number
Supported Script Types	All server scripts For more information, see the help topic SuiteScript 2.0 Script Types
Governance	20 units
Module	N/workflow Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.recordType	string	required	The record type ID of the workflow base record. Use values from the record.Type enum. This is the Record Type field on the Workflow Definition Page .	2015.2
options.recordId	string number	required	The internal ID of the base record.	2015.2
options.workflowId	string number	required	The internal ID (number) or script ID (string) for the workflow definition. This is the ID field on the Workflow Definition Page .	2015.2

Parameter	Type	Required / Optional	Description	Since
options.defaultValues	Object	optional	The object that contains key/value pairs to set default values on fields specific to the workflow. These can include fields on the Workflow Definition Page or workflow and state Workflow Custom Fields .	2015.2

Syntax

Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/workflow Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var workflowInstanceId = workflow.initiate({
4   recordType: 'customer',
5   recordId: 24,
6   workflowId: 'customworkflow_myWorkFlow'
7 });
8 ...
9 //Add additional code

```

workflow.trigger(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Triggers a workflow on a record. The actions and transitions of the workflow are evaluated for the record in the workflow instance, based on the current state for the workflow instance. Returns the internal ID of the workflow instance used to track the workflow against the record.
Returns	number
Supported Script Types	All server scripts For more information, see the help topic SuiteScript 2.0 Script Types
Governance	20 units
Module	N/workflow Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description	Since
options.recordType	string	required	The record type ID of the workflow base record. Use values from the record.Type	2015.2

Parameter	Type	Required / Optional	Description	Since
			enum. This is the Record Type field on the Workflow Definition Page .	
options.workflowId	string number	required	The internal ID (number) or script ID (string) for the workflow definition. This is the ID field on the Workflow Definition Page .	2015.2
options.workflowInstanceId	string number	optional	The internal ID of the workflow instance.	2015.2
options.actionId	string number	optional	The internal ID of a button that appears on the record in the workflow. Use this parameter to trigger the workflow as if the specified button were clicked.	2015.2
options.stateId	string number	optional	The internal ID (number) or script ID (string) of the workflow state that contains the action.	2015.2

Syntax



Important: The following code snippet shows the syntax for this member. It is not a functional example. For a complete script example, see [N/workflow Module Script Sample](#).

```

1 //Add additional code
2 ...
3 var workflowInstanceId = workflow.trigger({
4   recordType: 'salesorder',
5   workflowId: 'custworkflow_name',
6   actionId: workflowaction25
7 });
8 ...
9 //Add additional code

```

N/xml Module



Note: The content in this help topic pertains to SuiteScript 2.0.

Load the xml module to validate, parse, read, and modify XML documents.

- [N/xml Module Members](#)
- [Parser Object Members](#)
- [XPath Object Members](#)
- [Node Object Members](#)
- [Document Object Members](#)
- [Element Object Members](#)
- [Attr Object Members](#)
- [N/xml Module Script Samples](#)

N/xml Module Members

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Object	xml.Parser	Object	Client and server-side scripts	Encapsulates the functionality used by NetSuite to parse XML.
	xml.XPath	Object	Client and server-side scripts	Encapsulates the functionality used by NetSuite to run XPath expressions. XPath is a standard for enumerating paths in an XML document collection.
	xml.Node	Object	Client and server-side scripts	Represents a generic XML node in an XML document. A node can be a Document, Element, or Attribute.
	xml.Document	Object	Client and server-side scripts	Represents an entire XML document. The XML DOM presents a document as a hierarchy of node objects. Use the methods and properties available to the xml.Document object to manipulate the XML document and the nodes in the document tree.
	xml.Element	Object	Client and server-side scripts	Represents an element in an XML document. Elements may contain attributes, other elements, or text. If an element contains text, the text is represented in a text node of type TEXT_NODE.
	xml.Attr	Object	Client and server-side scripts	Represents an attribute node of an xml.Element object.
Method	xml.escape(options)	string	Client and server-side scripts	Prepares a string for use in XML by escaping XML markup, such as angle brackets, quotation marks, and ampersands.
	xml.validate(options)	void	Server-side scripts	Validates an XML document against an XML Schema (XSD).
Enum	xml.NodeType	string (read-only)	Client and server-side scripts	Enumeration that holds the string values for the supported node types. The Node.nodeType property is defined by one of the values in this enum.

Parser Object Members

The following members are called on the [xml.Parser](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Parser.fromString(options)	xml.Document	Client and server-side scripts	Parses a string into a W3C XML document object.
	Parser.toString(options)	string	Client and server-side scripts	Converts (serializes) an xml.Document object into a string.

XPath Object Members

The following members are called on the [xml.XPath](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	XPath.select(options)	xml.Node[]	Client and server-side scripts	Selects an array of nodes from an XML document using an XPath expression.

Node Object Members

The following members are called on the [xml.Node](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Node.appendChild(options)	xml.Node	Client and server-side scripts	Appends a node after the last child node of a specific element node. Returns the new child node.
	Node.cloneNode(options)	xml.Node	Client and server-side scripts	Creates a copy of a node. Returns the copied node.
	Node.compareDocumentPosition(options)	number	Client and server-side scripts	Returns a number that reflects where two nodes are located, compared to each other.
	Node.hasAttributes()	boolean true false	Client and server-side scripts	Returns true if the current node has child nodes or returns false if the current node does not have child nodes.
	Node.hasChildNodes()	boolean true false	Client and server-side scripts	Returns true if the current node has any attributes. Note that only element nodes can have attributes.
	Node.insertBefore(options)	xml.Node	Client and server-side scripts	Inserts a new child node before an existing child node for the current node.
	Node.isDefaultNamespace(options)	boolean true false	Client and server-side scripts	Returns true if the specified namespace uniform resource identifier (URI) is the default namespace for the current node or returns false if the specified namespace is not the default namespace.
	Node.isEqualNode(options)	boolean true false	Client and server-side scripts	Returns true if two nodes are equal or returns false if two nodes are not equal.
	Node.isSameNode(options)	boolean true false	Client and server-side scripts	Returns true if two nodes reference the same object or returns false if two nodes do not reference the same object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Node.lookupNamespaceURI(options)	string	Client and server-side scripts	Returns the namespace uniform resource identifier (URI) that matches the specified namespace prefix.
	Node.lookupPrefix(options)	string	Client and server-side scripts	Returns the namespace prefix associated with the specified namespace uniform resource identifier (URI).
	Node.normalize()	void	Client and server-side scripts	Puts all text nodes underneath a node, including attribute nodes, into a normal form.
	Node.removeChild(options)	xml.Node	Client and server-side scripts	Removes the specified child node. Returns the removed child node.
	Node.replaceChild(options)	xml.Node	Client and server-side scripts	Replaces a specific child node with another child node in a list of child nodes.
Property	Node.attributes	Object (read-only)	Client and server-side scripts	Key-value pairs for all attributes for an xml.Element node. Returns null for all other node types.
	Node.baseURI	string (read-only)	Client and server-side scripts	Absolute base uniform resource identifier (URI) of a node or null if the URI cannot be determined.
	Node.childNodes	xml.Node[] (read-only)	Client and server-side scripts	Array of all child nodes of a node or an empty array if there are no child nodes.
	Node.firstChild	xml.Node (read-only)	Client and server-side scripts	First child node for a specific node or null if there are no child nodes.
	Node.lastChild	xml.Node (read-only)	Client and server-side scripts	Last child node for a specific node or null if there is no last child node.
	Node.localName	string (read-only)	Client and server-side scripts	The local part of the qualified name of a node.
	Node.namespaceURI	string (read-only)	Client and server-side scripts	The namespace uniform resource identifier (URI) of a node or null if there is no namespace URI for the node.
	Node.nextSibling	xml.Node (read-only)	Client and server-side scripts	The next node in a node list or null if the current node is the last node.
	Node.nodeName	string (read-only)	Client and server-side scripts	Name of a node, depending on the type. For example, for a node of type xml.Element , the name is the name of the element.
	Node.nodeType	string	Client and server-side scripts	The type of node defined as a value from the xml.NodeType enum.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Node.nodeValue	string	Client and server-side scripts	The value of a node, depending on its type.
	Node.ownerDocument	xml.Document (read-only)	Client and server-side scripts	The root element for a node as a xml.Document object.
	Node.parentNode	xml.Node (read-only)	Client and server-side scripts	The parent node of a node.
	Node.prefix	string	Client and server-side scripts	The namespace prefix of the node, or null if the node does not have a namespace.
	Node.previousSibling	xml.Node (read-only)	Client and server-side scripts	The previous node in a node list or null if the current node is the first node.
	Node.textContent	string	Client and server-side scripts	The textual content of a node and its descendants.

Document Object Members

 **Note:** In addition to the Document object members, Document objects inherit the members of the Node object. The methods and properties associated with a Node object can be used as members of a Document object. For more information, see [Node Object Members](#).

The following members are called on the [xml.Document](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Document.adoptNode(options)	xml.Node	Client and server-side scripts	Attempts to adopt a node from another document to this document.
	Document.createAttribute(options)	xml.Attr	Client and server-side scripts	Creates an attribute node of type ATTRIBUTE_NODE with the optional specified value.
	Document.createAttributeNS(options)	xml.Attr	Client and server-side scripts	Creates an attribute node of type ATTRIBUTE_NODE, with the specified namespace value and optional specified value.
	Document.createCDATASection(options)	xml.Node	Client and server-side scripts	Creates a CDATA section node of type DOCUMENT_FRAGMENT_NODE with the specified data.
	Document.createComment(options)	xml.Node	Client and server-side scripts	Creates a Comment node of type COMMENT_NODE with the specified string.
	Document.createDocumentFragment()	xml.Node	Client and server-side scripts	Creates a node of type DOCUMENT_FRAGMENT_NODE.
	Document.createElement(options)	xml.Element	Client and server-side scripts	Creates a new node of type ELEMENT_NODE with the specified name.
	Document.createElementNS(options)	xml.Element	Client and server-side scripts	Creates a new node of type ELEMENT_NODE with the specified namespace URI and name.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
	Document.createProcessingInstruction(options)	xml.Node	Client and server-side scripts	Creates a new node of type PROCESSING_INSTRUCTION_NODE with the specified target and data.
	Document.createTextNode(options)	xml.Node	Client and server-side scripts	Creates a new node of type TEXT_NODE.
	Document.getElementById(options)	xml.Element	Client and server-side scripts	Returns the element that has an ID attribute with the specified value as an xml.Element object.
	Document.getElementsByTagName(options)	xml.Element[]	Client and server-side scripts	Returns an array of xml.Element objects with a specific tag name, in the order in which they appear in the XML document.
	Document.getElementsByTagNameNS(options)	xml.Element[]	Client and server-side scripts	Returns an array of xml.Element objects with a specific tag name and namespace, in the order in which they appear in the XML document.
	Document.importNode(options)	xml.Node	Client and server-side scripts	Imports a node from another document to this document. Creates a new copy of the source node.
Property	Document.doctype	Object (read-only)	Client and server-side scripts	Returns a node of type DOCUMENT_TYPE_NODE that represents the doctype of the XML document.
	Document.documentElement	xml.Element (read-only)	Client and server-side scripts	Root node of the XML document.
	Document.documentElementURI	string (read-only)	Client and server-side scripts	Location of the document or null if undefined.
	Document.inputEncoding	string (read-only)	Client and server-side scripts	Encoding used for an XML document at the time the document was parsed.
	Document.xmlEncoding	string (read-only)	Client and server-side scripts	Part of the XML declaration, the XML encoding of the XML document.
	Document.xmlStandalone	boolean true false	Client and server-side scripts	Part of the XML declaration, returns true if the current XML document is standalone or returns false if it is not.
	Document.xmlVersion	string	Client and server-side scripts	Part of the XML declaration, the version number of the XML document.

Element Object Members



Note: In addition to the Element object members, Element objects inherit the members of the Node object. The methods and properties associated with a Node object can be used as members of a Element object. For more information, see [Node Object Members](#).

The following members are called on the [xml.Element](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Method	Element.getAttribute(options)	string	Client and server-side scripts	Returns the value of the specified attribute.
	Element.getAttributeNode(options)	xml Attr	Client and server-side scripts	Retrieves an attribute node by name.
	Element.getAttributeNodeNS(options)	string	Client and server-side scripts	Returns an attribute node with the specified namespace URI and local name.
	Element.getAttributeNS(options)	xml Attr	Client and server-side scripts	Returns an attribute value with the specified namespace URI and local name.
	Element.getElementsByTagName(options)	xml Element[]	Client and server-side scripts	Returns an array of descendant <code>xml.Element</code> objects with a specific tag name, in the order in which they appear in the XML document.
	Element.getElementsByTagNameNS(options)	xml Element[]	Client and server-side scripts	Returns an array of descendant <code>xml.Element</code> objects with a specific tag name and namespace, in the order in which they appear in the XML document.
	Element.hasAttribute(options)	boolean true false	Client and server-side scripts	Returns <code>true</code> if the current element has an attribute with the specified name or if that attribute has a default value. Otherwise, returns <code>false</code> .
	Element.hasAttributeNS(options)	boolean true false	Client and server-side scripts	Returns <code>true</code> if the current element has an attribute with the specified local name and namespace or if that attribute has a default value. Otherwise, returns <code>false</code> .
	Element.removeAttribute(options)	void	Client and server-side scripts	Removes the attribute with the specified name.
	Element.removeAttributeNode(options)	xml Attr	Client and server-side scripts	Removes the attribute specified as a <code>xml Attr</code> object.
	Element.removeAttributeNS(options)	void	Client and server-side scripts	Removes the attribute with the specified namespace URI and local name.
	Element.setAttribute(options)	void	Client and server-side scripts	Adds a new attribute with the specified name. If an attribute with that name is already present in the element, its value is changed to the value specified in method argument.
	Element.setAttributeNode(options)	xml Attr	Client and server-side scripts	Adds the specified attribute node. If an attribute with the same name is already present in the element, it is replaced by the new one.
	Element.setAttributeNodeNS(options)	xml Attr	Client and server-side scripts	Adds the specified attribute node. If an attribute with the same local name and namespace URI is already present in the element, it is replaced by the new one.
	Element.setAttributeNS(options)	void	Client and server-side scripts	Adds a new attribute with the specified name and namespace URI. If an attribute with the same name and namespace URI is already present in the element, its value is changed to the value specified in method argument.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	Element.tagName	string (read-only)	Client and server-side scripts	The tag name of this xml.Element object.

Attr Object Members

The following members are called on the [xml.Attr](#) object.

Member Type	Name	Return Type / Value Type	Supported Script Types	Description
Property	Attr.name	string (read-only)	Client and server-side scripts	The name of an attribute.
	Attr.ownerElement	xml.Element (read-only)	Client and server-side scripts	The xml.Element object that is the parent of the xml.Attr object.
	Attr.specified	boolean true false	Client and server-side scripts	Returns true if the attribute value is set in the parsed XML document, and false if it is a default value in a DTD or Schema.
	Attr.value	string	Client and server-side scripts	Value of an attribute. The value of the attribute is returned as a string. Character and general entity references are replaced with their values.

N/xml Module Script Samples

The following script samples demonstrate how to use the features of the N/xml module. These samples reference the following XML file called BookSample.xml:

```

1 <bookstore xmlns:b="http://www.qualifiednamespace.com/book">
2   <b:book category="cooking">
3     <b:title lang="en">Everyday Italian</b:title>
4     <b:author>Giada De Laurentiis</b:author>
5     <b:year>2005</b:year>
6     <b:price>30.00</b:price>
7   </b:book>
8   <b:book category="children">
9     <b:title lang="en">Harry Potter</b:title>
10    <b:author>J. K. Rowling</b:author>
11    <b:year>2005</b:year>
12    <b:price>29.99</b:price>
13  </b:book>
14  <b:book category="web">
15    <b:title lang="en">XQuery Kick Start</b:title>
16    <b:author>James McGovern</b:author>
17    <b:author>Per Bothner</b:author>
18    <b:author>Kurt Cagle</b:author>
19    <b:author>James Linn</b:author>
20    <b:author>Vaidyanathan Nagarajan</b:author>
21    <b:year>2003</b:year>
22    <b:price>49.99</b:price>
23  </b:book>
24  <b:book category="web" cover="paperback">
25    <b:title lang="en">Learning XML</b:title>
26    <b:author>Erik T. Ray</b:author>
27    <b:year>2003</b:year>
28    <b:price>39.95</b:price>
29  </b:book>
```

30 | </bookstore>

Sample 1: Load an XML file and obtain child element values

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample loads the BookSample.xml file from the File Cabinet, iterates through the individual book nodes, and accesses the child node values.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet
4 */
5 require(['N/xml', 'N/file'], function(xml, file) {
6     return {
7         onRequest: function(options) {
8             var sentence = '';
9             var xmlFileContent = file.load('SuiteScripts/BookSample.xml').getContents();
10            var xmlDocument = xml.Parser.fromString({
11                text: xmlFileContent
12            });
13            var bookNode = xml.XPath.select({
14                node: xmlDocument,
15                xpath: '//b:book'
16            });
17
18            for (var i = 0; i < bookNode.length; i++) {
19                var title = bookNode[i].firstChild.nextSibling.textContent;
20                var author = bookNode[i].getElementsByTagName({
21                    tagName: 'b:author'
22                })[0].textContent;
23                sentence += 'Author: ' + author + ' wrote ' + title + '.\n';
24            }
25
26            options.response.write(sentence);
27        }
28    };
29 });

```

This script produces the following output when used with the BookSample.xml file:

```

1 Author: Giada De Laurentiis wrote Everyday Italian.
2 Author: J K. Rowling wrote Harry Potter.
3 Author: James McGovern wrote XQuery Kick Start.
4 Author: Erik T. Ray wrote Learning XML.

```

Sample 2: Parse an XML file and log element values

i Note: This sample script uses the require function so that you can copy it into the SuiteScript Debugger and test it. You must use the define function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following sample parses the XML string stored in the `xmlString` variable. The sample selects all config elements in the `xmlDocument` node, loops through them, and logs their contents.

```

1 /**
2  * @NApiVersion 2.x
3  * @NScriptType Suitelet

```

```

4  */
5
6 require(['N/xml'], function(xml) {
7     return {
8         onRequest: function(options) {
9             var xmlString = '<xml version="1.0" encoding="UTF-8"?><config date="1465467658668" transient="false">Some content</config>';
10
11         var xmlDoc = xml.Parser.fromString({
12             text: xmlString
13         });
14
15         var bookNode = xmlDoc.XPath.select({
16             node: xmlDoc,
17             xpath: '//config'
18         });
19
20         for (var i = 0; i < bookNode.length; i++) {
21             log.debug('Config content', bookNode[i].textContent);
22         }
23     };
24 };
25 });

```

Sample 3: Modify an XML file

Note: This sample script uses the `require` function so that you can copy it into the SuiteScript Debugger and test it. You must use the `define` function in an entry point script (the script you attach to a script record and deploy). For more information, see the help topics [SuiteScript 2.0 Script Basics](#) and [SuiteScript 2.0 Script Types](#).

The following samples shows how to modify an XML file.

```

1 /**
2 * @NApiVersion 2.x
3 */
4
5 require(['N/xml'], function(xml) {
6     var bookShelf = xml.Parser.fromString(file.load('SuiteScripts/books.xml').getContents());
7
8     var newBookNode = xmlData.createElement("book");
9     var newTitleNode = xmlData.createElement("title");
10    var newTitlenodeValue = xmlData.createTextNode("");
11    var newAuthorNode = xmlData.createElement("author");
12    var newAuthornodeValue = xmlData.createTextNode("");
13
14    newTitleNode.appendChild(newTitlenodeValue);
15    newAuthorNode.appendChild(newAuthornodeValue);
16    newBookNode.appendChild(newTitleNode);
17    newBookNode.appendChild(newAuthorNode);
18
19    var newbook = bookShelf.appendChild({
20        newChild: newBookNode
21    });
22 });

```

xml.Parser

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description

Encapsulates the functionality used by NetSuite to parse an XML document.

For a complete list of this object's methods, see [Parser Object Members](#).

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var parserObj = xml.Parser;
4 ...
5 //Add additional code

```

Parser.fromString(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Parses a String into a W3C XML document object. This API is useful if you want to navigate/query a structured XML document more effectively using either the Document API or NetSuite built-in XPath functions. Note: You can also use this method to validate your XML. If you pass a malformed string in as the options.text argument, Parser.fromString returns an SSS_XML_DOM_EXCEPTION error.
Returns	xml.Document
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.text	string	Required	String being converted to an xml.Document .

Errors

Error Code	Thrown If
SSS_XML_DOM_EXCEPTION	The input XML string is malformed.

Syntax

```
1 //Add additional code
```

```

2 ...
3 var xmlDocument = xml.Parser.fromString({
4   text : xmlStringContent
5 });
6 ...
7 //Add additional code

```

Parser.toString(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Converts (serializes) an xml.Document object into a string. This API is useful, for example, if you want to serialize and store an xml.Document in a custom field.
Returns	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.document	xml.Document	Required	XML document being serialized.

Syntax

```

1 //Add additional code
2 ...
3 var xmlStringContent = xml.Parser.toString({
4   document : xmlDocument
5 });
6 ...
7 //Add additional code

```

xml.XPath

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Encapsulates the functionality to run XPath expressions. For a complete list of this object's methods, see XPath Object Members .
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module

Since	2015.2
--------------	--------

Syntax

```

1 //Add additional code
2 ...
3 var xpath = xml.XPath;
4 ...
5 //Add additional code

```

XPath.select(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Selects an array of nodes from an XML that match an XPath expression.
Returns	xml.Node[]
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.node	xml.Node	Required	XML node being queried.
options.xpath	string	Required	XPath expression used to query node.

Syntax

```

1 //Add additional code
2 ...
3 var bookNode = xml.XPath.select({
4   node : xmlDoc,
5   xpath : '//book'
6 });
7 ...
8 //Add additional code

```

xml.Node

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Represents a single node in an XML document tree. The XML DOM presents a document as a hierarchy of node objects. See the xml.NodeType enum for a list of possible node types.
---------------------------	--

	<p>You can use this object to work with a child node, or nested nodes.</p> <p>NetSuite supports a subset of W3C DOM methods. For a complete list of this object's methods and properties, see Node Object Members.</p> <p>For other code snippets that use this object, see the syntax sample that follows, as well as Node.childNodes and N/xml Module Script Sample.</p>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var bookNode = xml.XPath.select({
4   node : xmlDocument,
5   xpath : '//book'
6 });
7 ...
8 //Add additional code

```

Node.appendChild(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Appends a node after the last child node of a specific element node. Returns the new child node.
Returns	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.newChild	xml.Node	Required	xml.Node object to append.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	HIERARCHY_REQUEST_ERR: An attempt was made to insert a node where it is not permitted.	Node cannot be appended.

Syntax

```

1 //Add additional code
2 ...
3 var bookShelf = xml.Parser.fromString(file.load('SuiteScripts/books.xml').getContents());
4
5 var newBookNode = xmlData.createElement("book");
6 var newTitleNode = xmlData.createElement("title");
7 var newTitlenodeValue = xmlData.createTextNode("");
8 var newAuthorNode = xmlData.createElement("author");
9 var newAuthornodeValue = xmlData.createTextNode("");
10 newTitleNode.appendChild(newTitlenodeValue);
11 newAuthorNode.appendChild(newAuthornodeValue);
12 newBookNode.appendChild(newTitleNode);
13 newBookNode.appendChild(newAuthorNode);
14
15 var newbook = bookShelf.appendChild({
16     newChild : newBookNode
17 });
18 ...
19 //Add additional code

```

Node.cloneNode(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a copy of a node. Returns the copied node.
Returns	<code>xml.Node</code>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.deep</code>	<code>boolean true false</code>	Optional	Use <code>true</code> to clone the node, attributes, and all descendants. Use <code>false</code> to only clone the node and attributes.

Syntax

```

1 //Add additional code
2 ...
3 var copiednode = elem[0].cloneNode({
4     deep : true
5 });
6 ...
7 //Add additional code

```

Node.compareDocumentPosition(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns a number that reflects where two nodes are located, compared to each other. Returns one of the following numbers: <ul style="list-style-type: none">■ 1. The two nodes do not belong to the same document.■ 2. The specified node comes before the current node.■ 4. The specified node comes after the current node.■ 8. The specified node contains the current node.■ 16. The current node contains the specified node.■ 32. The specified and current nodes do not have a common container node or the two nodes are different attributes of the same node.
	Note: The return value can be a combination of the above values. For example, a return value of 20 means the specified node is contained by the current node, a value of 16, and the specified node follows the current node, a value of 4.
	Important: This method is not supported on Internet Explorer.
Returns	number
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.other	xml.Node	Required	The node to compare with the current node.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected xml.Node or subclass: other	The options.other is of type xml.Node .

Syntax

```

1 //Add additional code
2 ...
3 var posCode = elem[0].compareDocumentPosition({}
```

```

4     other : parentNode[0]
5   });
6   ...
7 //Add additional code

```

Node.hasAttributes()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the current node has attributes defined, or false otherwise.
	Important: This method is not supported on Internet Explorer.
Returns	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var hasAttributes = parentNode[0].hasAttributes()
4 ...
5 //Add additional code

```

Node.hasChildNodes()



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the current node has child nodes or returns false if the current node does not have child nodes.
Returns	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code

```

```

2 ...
3 var hasChildren = parentNode[0].hasChildNodes()
4 ...
5 //Add additional code

```

Node.insertBefore(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Inserts a new child node before an existing child node for the current node. If the new child node is already in the list of children, this method removes the new child node and inserts it again.
Returns	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.newChild	xml.Node	Required	The new child node to insert.
options.refChild	xml.Node	Required	The node before which to insert the new child node. If refChild is , the method inserts the new node at the end of the list of children.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	HIERARCHY_REQUEST_ERR: An attempt was made to insert a node where it is not permitted.	Node cannot be inserted.

Syntax

```

1 //Add additional code
2 ...
3 var insertednode = parentNode[0].insertBefore({
4   newChild : elemlist1[0],
5   refChild : elemlist2[0]
6 });
7 ...
8 //Add additional code

```

Node.isDefaultNamespace(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the specified namespace uniform resource identifier (URI) is the default namespace for the current node or returns false if the specified namespace is not the default namespace. See also Node.namespaceURI .
Returns	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	The namespace URI to compare.

Syntax

```

1 //Add additional code
2 ...
3 var isDefault = parentNode[0].isDefaultNamespace({
4   namespaceURI : '*'
5 });
6 ...
7 //Add additional code

```

Node isEqualNode(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if two nodes are equal or returns false if two nodes are not equal. The two nodes are equal if they meet the following conditions: <ul style="list-style-type: none"> ■ Both nodes have the same type. ■ Both nodes have the same attributes and attribute values. The order of the attributes is not considered. ■ Both nodes have equal lists of child nodes and the child nodes appear in the same order.
---------------------------	---

	<p>Note: Two nodes may be equal, even if they are not the same. See Node.isSameNode(options).</p>
	<p>Important: This method is not supported on Internet Explorer.</p>
Returns	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description
options.other	xml.Node	Required	The node to compare with the current node.

Syntax

```

1 //Add additional code
2 ...
3 var isEqual = elem[0].isEqualNode({
4     other : node
5 });
6 ...
7 //Add additional code

```

Node.isSameNode(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Returns true if two nodes reference the same object or returns false if two nodes do not reference the same object. If two nodes are the same, all attributes have the same values and you can use methods on the two nodes interchangeably.
	<p>Note: Two nodes that are the same are also equal. See Node.isEqualNode(options).</p>
	<p>Important: This method is not supported on Internet Explorer or Firefox.</p>
Returns	boolean true false
Supported Script Types	All script types

	For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.other	xml.Node	Required	The node to compare with the current node.

Syntax

```

1 //Add additional code
2 ...
3 var isSame = elem[0].isSameNode({
4     other : node
5 });
6 ...
7 //Add additional code

```

Node.lookupNamespaceURI(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the namespace uniform resource identifier (URI) that matches the specified namespace prefix. Returns null if the specified prefix does not have an associated URI.
	Important: This method is not supported on Internet Explorer.
Returns	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.prefix	string	Required	Namespace prefix associated with the namespace URI.

Syntax

```

1 //Add additional code
2 ...
3 var uri = parentNode[0].lookupNamespaceURI({
4     prefix : '*'
5 });
6 ...
7 //Add additional code

```

Node.lookupPrefix(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the namespace prefix associated with the specified namespace uniform resource identifier (URI). Returns null if the specified URI does not have an associated prefix. If more than one prefix is associated with the namespace prefix, the namespace returned by this method depends on the module implementation.
Returns	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI associated the namespace prefix.

Syntax

```

1 //Add additional code
2 ...
3 var prefix = parentNode[0].lookupPrefix({
4     namespaceURI : '*'
5 });
6 ...
7 //Add additional code

```

Node.normalize()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Puts all text nodes underneath a node, including attribute nodes, into a normal form. In normal form, only structure (such as elements, comments, processing instructions, CDATA sections, and entity references) separates text nodes. After normalization, there are no adjacent or empty text nodes. Use this method if you require a particular document tree structure and want to make sure that the XML DOM view of a document is identical when you save and reload it.
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 node.normalize();
4 ...
5 //Add additional code

```

Node.removeChild(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the specified child node.
Returns	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.oldChild	xml.Node	Required	Node to remove.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	NOT_FOUND_ERR: An attempt is made to reference a node in a context where it does not exist.	Node cannot be removed.

Syntax

```

1 //Add additional code
2 ...
3 var removednode = parentNode[0].removeChild({
4     oldChild : node
5 });
6 ...
7 //Add additional code

```

Node.replaceChild(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Replaces a specific child node with another child node in a list of child nodes. If the new child node to add already exists in the list of child nodes, the node is first removed.
Returns	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.newChild	xml.Node	Required	New child node to add.
options.oldChild	xml.Node	Required	Child node to replaced with the new node.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	NOT_FOUND_ERR: An attempt is made to reference a node in a context where it does not exist.	Child node cannot be found.

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	HIERARCHY_REQUEST_ERR: An attempt was made to insert a node where it is not permitted.	Child node cannot be replaced.

Syntax

```

1 //Add additional code
2 ...
3 var replacednode = parentNode.replaceChild({
4     newChild : elem[2],
5     oldChild : elem[1]
6 });
7 ...
8 //Add additional code

```

Node.attributes

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Key-value pairs for all attributes for an xml.Element node. Returns null for all other node types.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var attrbs = elem[0].attributes;
4 ...
5 //Add additional code

```

Node.baseURI

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Absolute base uniform resource identifier (URI) of a node or null if the URI cannot be determined. For client scripts, this property always returns null.
Type	string (read-only)

Note: The format of this value is browser-specific.

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var baseuri = parentNode[0].baseURI;
4 ...
5 //Add additional code

```

Node.childNodes

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Array of all child nodes of a node or an empty array if there are no child nodes.
Type	xml.Node[]
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var childnodes = parentNode[0].childNodes;
4 ...
5 //Add additional code

```

Node.firstChild

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The first child node of a node, or null if there are no child nodes.
Type	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var nodeValue1 = bookNode[0].firstChild.nextSibling.textContent;
4 ...
5 //Add additional code

```

Node.lastChild

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The last child node of a node, or null if there are no child nodes.
Type	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var nodeValue = parentNode[0].lastChild.previousSibling.textContent;
4 ...
5 //Add additional code

```

Node.localName

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Property Description	The local part of the qualified name of a node.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var localname = parentNode[0].localName;
4 ...
5 //Add additional code

```

Node.namespaceURI

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The namespace uniform resource identifier (URI) of a node or null if there is no namespace URI for the node.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var uri = parentNode[0].namespaceURI;
4 ...
5 //Add additional code

```

Node.nextSibling

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The next node in a node list or null if the current node is the last node.
Type	xml.Node (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var nodeName = parentNode[0].firstChild.nextSibling.textContent;
4 ...
5 //Add additional code

```

Node.nodeName

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Name of a node, depending on the type. For example, for a node of type xml.Element , the name is the name of the element.
-----------------------------	---

	Note: On Chrome, this property also includes the namespace or prefix.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var nodeName = parentNode[0].firstChild.nodeName;
4 ...
5 //Add additional code

```

Node.nodeType

Note: The content in this help topic pertains to SuiteScript 2.0.	
Property Description	The type of node as an enum. For all possible values of this property, see xml.NodeType .
Type	xml.NodeType
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var nodeType = parentNode[0].firstChild.nodeType;
4 ...
5 //Add additional code

```

Node.nodeValue

Note: The content in this help topic pertains to SuiteScript 2.0.	
Property Description	The value of a node, depending on its type. If the value is null, setting this value has no effect.
Type	string

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var nodeValue = parentNode[0].firstChild.nodeValue;
4 ...
5 //Add additional code

```

Node.ownerDocument

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The root element for a node as a xml.Document object. Use this object to create new nodes with Document.createElement(options) or Document.createElementNS(options) .
Type	xml.Document
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var doc = parentNode[0].ownerDocument;
4 ...
5 //Add additional code

```

Node.parentNode

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The parent node of a node. All node types, except xml.Attr , xml.Document , DocumentFragment , Entity , and Notation can have a parent node. See xml.NodeType for possible node types.
Type	xml.Node
Supported Script Types	All script types

	For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code ...
2 var nodevalue = parentNode[0].lastChild.parentNode.textContent;
3 ...
4 //Add additional code

```

Node.prefix

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The namespace prefix of the node, or null if the node does not have a namespace. If the value is null, setting it has no effect, including read-only node types.
Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	NAMESPACE_ERR: An attempt is made to create or change an object in a way which is incorrect with regard to namespaces.	Cannot edit the node prefix.

Syntax

```

1 //Add additional code
2 ...
3 var namespacePrefix = parentNode[0].firstChild.prefix;
4 ...
5 //Add additional code

```

Node.previousSibling

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The previous node in a node list or null if the current node is the first node.
-----------------------------	---

Type	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var nodeValue = parentNode[0].lastChild.previousSibling.textContent;
4 ...
5 //Add additional code

```

Node.textContent

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The textual content of a node and its descendants. If the value is null, then setting it has no effect. If you set this value, any child nodes are removed and replaced by a single text node with this string as a value.
Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var nodeValue = parentNode[0].firstChild.textContent;
4 ...
5 //Add additional code

```

xml.Document

Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Represents an entire XML document. The XML DOM presents a document as a hierarchy of node objects. Use the methods and properties available to the xml.Document object to manipulate the XML document and the nodes in the document tree. For a list of this object's methods and properties, see Document Object Members .
---------------------------	--

	An XML document object is also a node of type DOCUMENT_NODE. In addition to the Document object members, Document objects inherit the members of the Node object. For a complete list of these methods and properties, see Node Object Members .
Supported Script Types	All script types
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var xmlDocument = xml.Parser.fromString({
4   text : xmlFileContent
5 });
6 ...
7 //Add additional code

```

Document.adoptNode(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Attempts to adopt a node from another document to this document. If successful, this method changes the Node.ownerDocument property of the source node, its children, and any attribute nodes to the current document. If the source node has a parent node, the parent node is first removed from the child list of its own parent node.
	 Important: This method is not supported on Internet Explorer.
Returns	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.source	xml.Node	Required	Source node to add as a child into the current node object.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	NOT_FOUND_ERR: An attempt is made to reference a node in a context where it does not exist.	Node cannot be adopted.

Syntax

```

1 //Add additional code
2 ...
3 var adoptedNode = xmlDocument1.adoptNode({
4     source : sourceNode,
5 });
6 ...
7 //Add additional code

```

Document.createAttribute(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates an attribute node of type ATTRIBUTE_NODE with the optional specified value and returns the new xml.Attr object. The localName, prefix, and namespaceURI properties of the new node are set to null.
Returns	xml.Attr
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	Required	Name of the new attribute node.
options.value	string	Optional	Value for the attribute node. If unspecified, the value is an empty string.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INVALID_CHARACTER_ERR: An invalid or illegal XML character is specified.	Attribute with the specified name or value cannot be created.

Syntax

```

1 //Add additional code
2 ...
3 var attr = xmlDocument.createAttribute({
4     name : 'lang',
5     value : 'fr'
6 });
7 ...
8 //Add additional code

```

Document.createAttributeNS(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates an attribute node of type ATTRIBUTE_NODE, with the specified namespace value and optional specified value, and returns the new xml.Attr object. The Node.localName , Node.prefix , and Node.namespaceURI properties of the new node are set to null.
Returns	xml.Attr
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI of the attribute to create. Value can be null.
options.qualifiedName	string	Required	Qualified name of the new attribute node.
options.value	string	Optional	Value for the attribute node. If unspecified, the value is an empty string.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INVALID_CHARACTER_ERR: An invalid or illegal XML character is specified.	Attribute with the specified value cannot be created.

Syntax

```

1 //Add additional code
2 ...
3 var attr = xmlDocument.createAttributeNS({
4   namespaceURI : '*',
5   qualifiedName : 'lang',
6   value : 'fr'
7 });
8 ...
9 //Add additional code

```

Document.createCDATASection(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a CDATA section node of type DOCUMENT_FRAGMENT_NODE with the specified data and returns the new xml.Node object.
Returns	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.data	string	Required	Data for the new CDATA section node.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected string: data	Cannot create CDATA section node with the specified data.

Syntax

```

1 //Add additional code
2 ...
3 var newNode = xmlDocument.createCDATASection({
4   data : 'Limited Edition.'
5 });
6 ...
7 //Add additional code

```

Document.createComment(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a Comment node of type COMMENT_NODE with the specified string.
Returns	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.data	string	Required	Data for the Comment node.

Syntax

```

1 //Add additional code
2 ...
3 var newNode = xmlDocument.createComment({
4   data : 'This is a comment.'
5 });
6 ...
7 //Add additional code

```

Document.createDocumentFragment()

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a node of type DOCUMENT_FRAGMENT_NODE and returns the new xml.Node object.
Returns	xml.Node
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var newNode = xmlDocument.createDocumentFragment();
4 ...
5 //Add additional code

```

Document.createElement(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new node of type ELEMENT_NODE with the specified name and returns the new xml.Element node. The Node.localName , Node.prefix , and Node.namespaceURI properties of the new node are set to null.
Returns	xml.Element
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters



Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.tagName	string	Required	Name of the element to create.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INVALID_CHARACTER_ERR: An invalid or illegal XML character is specified.	Element cannot be created with the specified tagName value.

Syntax

```

1 //Add additional code
2 ...
3 var elem = xmlDocument.createElement({
4   tagName : 'book'
5 });
6 ...
7 //Add additional code

```

Document.createElementNS(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new node of type ELEMENT_NODE with the specified namespace URI and name and returns the new xml.Element object. The Node.localName , Node.prefix , and Node.namespaceURI properties of the new node are set to null.
Returns	xml.Element
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI of the element to create. Can be null.
options.qualifiedName	string	Required	Qualified name of the element to create.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INVALID_CHARACTER_ERR: An invalid or illegal XML character is specified.	Element with the specified namespace cannot be created.

Syntax

```

1 //Add additional code
2 ...
3 var elem = xmlDocument.createElementNS({
4   namespaceURI : '*',
5   qualifiedName : 'book'
6 });
7 ...
8 //Add additional code

```

Document.createProcessingInstruction(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new node of type PROCESSING_INSTRUCTION_NODE with the specified target and data and returns the new xml.Node object.
---------------------------	--

	<p>The following example shows a sample processing instruction:</p> <pre><?xml version="1.0"?></pre> <p>Use a processing instruction node to keep processor-specific information in the text of the XML document.</p>
Returns	<code>xml.Node</code>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.target	string	Required	Target part of the processing instruction.
options.data	string	Required	Data for the processing instruction.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INVALID_CHARACTER_ERR: An invalid or illegal XML character is specified.	Processing instruction node cannot be created with the specified target or data.

Syntax

```

1 //Add additional code
2 ...
3 var newNode = xmlDocument.createProcessingInstruction(
4     target : 'xml'
5     data : 'version="1.0"'
6 );
7 ...
8 //Add additional code

```

Document.createTextNode(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Creates a new text node and returns the new <code>xml.Node</code> object.
Returns	<code>xml.Node</code>

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.data	string	Required	Data for the text node.

Syntax

```

1 //Add additional code
2 ...
3 var newNode = xmlDoc.createElement({
4   data : 'Sample Title'
5 });
6 ...
7 //Add additional code

```

Document.getElementById(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the element that has an ID attribute with the specified value as an xml.Element object. Returns null if no such element exists.
Returns	xml.Element
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.elementId	string	Required	Unique ID value for an element.

Syntax

```

1 //Add additional code
2 ...
3 var elem = xmlDocument.getElementById({
4   elementId : 'id12345'
5 });
6 ...
7 //Add additional code

```

Document.getElementsByTagName(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an array of xml.Element objects with a specific tag name, in the order in which they appear in the XML document.
Returns	xml.Element[]
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.tagName	string	Required	Case-sensitive tag name of the element to match on. Use the * wildcard to match all elements.

Syntax

```

1 //Add additional code
2 ...
3 var elem = xmlDocument.getElementsByTagName({
4   tagName : 'book'
5 });
6 ...
7 //Add additional code

```

Document.getElementsByTagNameNS(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an array of xml.Element objects with a specific tag name and namespace, in the order in which they appear in the XML document.
---------------------------	--

	Important: This method is not supported on Internet Explorer.
Returns	<code>xml.Element[]</code>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description
<code>options.namespaceURI</code>	string	Required	Namespace URI to match on. Use the * wildcard to match all namespaces.
<code>options.localName</code>	string	Required	Localname property to match on. Use the * wildcard to match all local names.

Syntax

```

1 //Add additional code
2 ...
3 var elem = xmlDocument.getElementsByTagNameNS(
4   namespaceURI : '*',
5   localName : 'book'
6 );
7 ...
8 //Add additional code

```

Document.importNode(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Imports a node from another document to this document. This method creates a new copy of the source node. If the deep parameter is set to true, it imports all children of the specified node. If set to false, it imports only the node itself. Method returns the imported <code>xml.Node</code> object.
Returns	<code>xml.Node</code>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description
options.importedNode	xml.Node	Required	Node from another XML document to import.
options.deep	boolean true false	Required	Use true to import the node, its attributes, and all descendants. Use false to only import the node and its attributes.



Important: This parameter is not supported on Internet Explorer.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	NOT_SUPPORTED_ERR: The implementation does not support the requested type of object or operation.	Node cannot be imported.

Syntax

```

1 ...
2 var importedNode = xmlDocument1.importNode({
3   importedNode : foreignNode,
4   deep : true
5 });
6 ...

```

Document.doctype

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Property Description	The doctype of the XML document.
	Important: This property is not supported on Internet Explorer.
Type	xml.Element (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var doctype = xmlDocument.doctype;
4 ...
5 //Add additional code

```

Document.documentElement

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Root node of the XML document. Use this property to directly access the xml.Element object that represents the root node of an XML document.
Type	xml.Element (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var root = xmlDocument.documentElement;
4 ...
5 //Add additional code

```

Document.documentElementURI

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Location of the document or null if undefined.
Type	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var documentURI = xmlDocument.documentElementURI;

```

```

4 | ...
5 | //Add additional code

```

Document.inputEncoding

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Encoding used for an XML document at the time the document was parsed. When parsing an XML document with the following declaration, the inputEncoding property is UTF-8: <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 | //Add additional code
2 | ...
3 | var encoding = xmlDocument.inputEncoding;
4 | ...
5 | //Add additional code

```

Document.xmlEncoding

Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Part of the XML declaration, the XML encoding of the XML document. In the following declaration, the xmlEncoding property is UTF-8: <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var encoding = xmlDocument.xmlEncoding;
4 ...
5 //Add additional code

```

Document.xmlStandalone



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Part of the XML declaration, returns true if the current XML document is standalone or returns false if it is not. In the following declaration, the xmlStandalone property is true: <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
Type	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var isStandalone = xmlDocument.xmlStandalone;
4 ...
5 //Add additional code

```

Document.xmlVersion



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Part of the XML declaration, the version number of the XML document. In the following declaration, the xmlVersion property is 1.0: <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
Type	string (read-only)

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	The implementation does not support the requested type of object or operation.	Cannot edit the XML version for the document.

Syntax

```

1 //Add additional code
2 ...
3 var version = xmlDocument.xmlVersion;
4 ...
5 //Add additional code

```

xml.Element

Note: The content in this help topic pertains to SuiteScript 2.0.	
Object Description	Represents an element in an XML document. Elements may contain attributes, other elements, or text. If an element contains text, the text is represented in a text node of type TEXT_NODE. For example, the following element year contains a text node with the value of 2015: <year>2015</year> For a list of this object's methods and properties, see Element Object Members An XML element object is also a node of type ELEMENT_NODE. In addition to the Element object members, Element objects inherit the members of the Node object. For a complete list of these methods and properties, see Node Object Members .
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var elem = parentNode[0].getElementsByTagName({tagName : 'title'});
4 ...
5 //Add additional code

```

Element.getAttribute(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns the value of the specified attribute.
Returns	xml.Attr
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	Required	Name of the attribute for which to return the value.

Syntax

```

1 //Add additional code
2 ...
3 var attr = elem[0].getAttribute({
4   name : 'lang'
5 });
6 ...
7 //Add additional code

```

Element.getAttributeNode(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Retrieves an attribute node by name. <div style="background-color: #ffffcc; padding: 5px;">  Important: This method is not supported on Internet Explorer. </div>
Returns	xml.Attr
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	Required	The name of the attribute to return.

Syntax

```

1 //Add additional code
2 ...
3 var attr = elem[0].getATTRIBUTE_NODE({
4   name : 'lang'
5 });
6 ...
7 //Add additional code

```

Element.getAttributeNodeNS(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an attribute node with the specified namespace URI and local name.
	 Important: This method is not supported on Internet Explorer.
Returns	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI of the attribute to return. Value can be null.
options.localName	string	Required	Local name of the attribute to return.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected string: namespaceURI	Attribute node with the specified namespace cannot be retrieved.

Syntax

```

1 //Add additional code
2 ...
3 var attr = elem[0].getgetAttributeNodeNS({
4   namespaceURI : '*',
5   localName : 'lang'
6 });
7 ...
8 //Add additional code

```

Element.getAttributeNS(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an attribute value with the specified namespace URI and local name.  Important: This method is not supported on Internet Explorer.
Returns	string
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI of the attribute to return. Value can be null.
options.localName	string	Required	Local name of the attribute to return.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected string: namespaceURI	Attribute with the specified namespace cannot be retrieved.

Syntax

```

1 //Add additional code
2 ...
3 var attr = elem[0].getgetAttributeNS({

```

```

4     namespaceURI : '*'
5     localName : 'lang'
6   });
7   ...
8 //Add additional code

```

Element.getElementsByTagName(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an array of descendant <code>xml.Element</code> objects with a specific tag name, in the order in which they appear in the XML document.
Returns	<code>xml.Element[]</code>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
<code>options.tagName</code>	string	Required	Case-sensitive tag name of the element to match on. Use the <code>*</code> wildcard to match all elements.

Syntax

```

1 //Add additional code
2 ...
3 var elem = parentNode[0].getElementsByTagName({tagName : 'title'});
4 ...
5 //Add additional code

```

Element.getElementsByTagNameNS(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns an array of descendant <code>xml.Element</code> objects with a specific tag name and namespace, in the order in which they appear in the XML document.
Returns	<code>xml.Element[]</code>



Important: This method is not supported on Internet Explorer.

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI to match on. Use the * wildcard to match all namespaces.
options.localName	string	Required	Localname property to match on. Use the * wildcard to match all local names.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected string: namespaceURI	Elements with the specified namespace cannot be retrieved.

Syntax

```

1 //Add additional code
2 ...
3 var elem = parentNode[0].getElementsByTagNameNS({
4   namespaceURI : '*',
5   localName : 'lang'
6 });
7 ...
8 //Add additional code

```

Element.hasAttribute(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the current element has an attribute with the specified name or if that attribute has a default value. Otherwise, returns false.
	 Important: This method is not supported on Internet Explorer.
Returns	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	Required	Name of the attribute to match on.

Syntax

```

1 //Add additional code
2 ...
3 var attrExists = elem[0].hasAttribute({
4   name : 'lang'
5 });
6 ...
7 //Add additional code

```

Element.hasAttributeNS(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Returns true if the current element has an attribute with the specified local name and namespace or if that attribute has a default value. Otherwise, returns false.
	Important: This method is not supported on Internet Explorer.
Returns	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI of the attribute to match on.
options.localName	string	Required	Local name of the attribute to match on.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected string: namespaceURI	The method is called with an illegal namespace value.

Syntax

```

1 //Add additional code
2 ...
3 var attrExists = elem[0].hasAttributeNS({
4     namespaceURI : '*',
5     localName : 'lang'
6 });
7 ...
8 //Add additional code

```

Element.removeAttribute(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the attribute with the specified name.
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.name	string	Required	Name of the attribute to remove.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected string: name	Attribute with the specified name cannot be removed.

Syntax

```

1 //Add additional code
2 ...

```

```

3 elem[0].removeAttribute({
4   name : 'lang'
5 });
6 ...
7 //Add additional code

```

Element.removeAttributeNode(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the attribute specified as a xml.Attr object.
Returns	xml.Attr
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.oldAttr	xml.Attr	Required	xml.Attr object to remove.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	NOT_FOUND_ERR: An attempt is made to reference a node in a context where it does not exist.	Attribute node cannot be removed.

Syntax

```

1 //Add additional code
2 ...
3 var removedAttr = elem[0].removeAttributeNode({
4   oldAttr : attr
5 });
6 ...
7 //Add additional code

```

Element.removeAttributeNS(options)

Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Removes the attribute with the specified namespace URI and local name.
---------------------------	--

	 Important: This method is not supported on Internet Explorer.
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI of the attribute node to remove.
options.localName	string	Required	Local name of the attribute node to remove.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected string: namespaceURI	Attribute with the specified namespace cannot be removed.

Syntax

```

1 //Add additional code
2 ...
3 elem[0].removeAttributeNS({
4   namespaceURI : '*',
5   localName : 'lang'
6 });
7 ...
8 //Add additional code

```

Element.setAttribute(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds a new attribute with the specified name. If an attribute with that name is already present in the element, its value is changed to the value specified in method argument. If an attribute with the specified name already exists, the value of the attribute is changed to the value of the value parameter.
Returns	void

Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note:	The options parameter is a JavaScript object.
--------------	---

Parameter	Type	Required / Optional	Description
options.name	string	Required	Name of the attribute to add.
options.value	string	Required	Value of the attribute to add.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INVALID_CHARACTER_ERR: An invalid or illegal XML character is specified.	Value for the attribute cannot be set.

Syntax

```

1 //Add additional code
2 ...
3 elem[0].setAttribute({
4   name : 'Lang',
5   value : 'fr'
6 });
7 ...
8 //Add additional code

```

Element.setAttributeNode(options)

Note:	The content in this help topic pertains to SuiteScript 2.0.
--------------	---

Method Description	Adds the specified attribute node. If an attribute with the same name is already present in the element, it is replaced by the new one. If an attribute with the same nodeName property already exists, it is replaced with the object in the newAttr parameter. If the attribute node replaces an existing attribute node, the method returns the new xml.Attr object.
Returns	xml.Attr
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .

Governance	None
Module	N/xml Module
Since	2015.2

Parameters

 **Note:** The options parameter is a JavaScript object.

Parameter	Type	Required / Optional	Description
options.newAttr	xml.Attr	Required	New <code>xml.Attr</code> object to add to the <code>xml.Element</code> object.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INUSE_ATTRIBUTE_ERR: An attempt is made to add an attribute that is already in use elsewhere.	Attribute node cannot be added.

Syntax

```

1 //Add additional code
2 ...
3 elem[0].setAttributeNode({
4   newAttr : attr
5 });
6 ...
7 //Add additional code

```

Element.setAttributeNodeNS(options)

 **Note:** The content in this help topic pertains to SuiteScript 2.0.

Method Description	Adds the specified attribute node. If an attribute with the same local name and namespace URI is already present in the element, it is replaced by the new one. If an attribute with the same namespaceURI and localName property already exist, it is replaced with the object in the newAttr parameter. If the attribute node replaces an existing attribute node, the method returns the new <code>xml.Attr</code> object.
Returns	<code>xml.Attr</code>
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module

Since	2015.2
--------------	--------

Parameters

 Note:	The options parameter is a JavaScript object.
--	---

Parameter	Type	Required / Optional	Description
options.newAttr	xml.Attr	Required	New xml.Attr object to add to the xml.Element object.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INUSE_ATTRIBUTE_ERR: An attempt is made to add an attribute that is already in use elsewhere.	Attribute node cannot be added.

Syntax

```

1 //Add additional code
2 ...
3 elem[0].setAttributeNode({
4   newAttr : attr
5 });
6 ...
7 //Add additional code

```

Element.setAttributeNS(options)

 Note:	The content in this help topic pertains to SuiteScript 2.0.
--	---

Method Description	Adds a new attribute with the specified name and namespace URI. If an attribute with the same name and namespace URI is already present in the element, its value is changed to the value specified in method argument. If an attribute with the specified name already exists, the value of the attribute is changed to the value of the value parameter. If the attribute node replaces an existing attribute node, the method returns the new xml.Attr object.
Returns	void
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note:	The options parameter is a JavaScript object.
--------------	---

Parameter	Type	Required / Optional	Description
options.namespaceURI	string	Required	Namespace URI of the attribute node to add.
options.qualifiedName	string	Required	Fully qualified attribute name to add.
options.value	string	Required	String value of the attribute to add.

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	INVALID_CHARACTER_ERR: An invalid or illegal XML character is specified.	Attribute node with the specified value cannot be added.

Syntax

```

1 //Add additional code
2 ...
3 elem[0].setAttributeNS({
4   namespaceURI : '*',
5   qualifiedName : 'lang',
6   value : 'fr'
7 });
8 ...
9 //Add additional code

```

Element.tagName

Note:	The content in this help topic pertains to SuiteScript 2.0.
--------------	---

Property Description	The tag name of this <code>xml.Element</code> object.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var tagName = elem[0].tagName; \\ returns 'title'.
4 ...

```

```
5 //Add additional code
```

xml.Attr



Note: The content in this help topic pertains to SuiteScript 2.0.

Object Description	Represents an attribute node of an xml.Element object. For a complete list of this object's properties, see Attr Object Members .
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```
1 //Add additional code
2 ...
3 var attr = elem[0].getATTRIBUTENode({
4   name : 'lang'
5 });
6 ...
7
8 //Add additional code
```

Attr.name



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	The name of an attribute. This property is a qualified name if the Node.localName property for the parent xml.Element object is null.
Type	string (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```
1 //Add additional code
2 ...
3 var attrName = attr.name; \\ returns 'lang'.
4 ...
```

```
5 //Add additional code
```

Attr.ownerElement



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<code>xml.Element</code> object that is the parent of the <code>xml.Attr</code> object. Value is null if the attribute is not used by an element.
	Important: This property is not supported on Internet Explorer.
Type	<code>xml.Element</code> (read-only)
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```
1 //Add additional code
2 ...
3 var attrElement = attr.ownerElement; \\ returns the title element.
4 ...
5 //Add additional code
```

Attr.specified



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	Returns true if the attribute value is set in the parsed XML document, and false if it is a default value in a DTD or Schema.
Type	boolean true false
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Syntax

```
1 //Add additional code
2 ...
3 var attrSpecified = attr.specified;
4 ...
5 //Add additional code
```

Attr.value



Note: The content in this help topic pertains to SuiteScript 2.0.

Property Description	<p>Value of an attribute. The value of the attribute is returned as a string. Character and general entity references are replaced with their values. For example, a character reference such as &#160; or an entity reference such as &nbsp; is replaced with a non-breaking space.</p> <p>Note: If you set this value, it creates a text node with the unparsed contents of the string, for example, any characters that an XML processor would recognize as markup are instead treated as literal text.</p>
Type	string
Supported Script Types	<p>All script types For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Module	N/xml Module
Since	2015.2

Errors

Error Code	Message	Thrown If
SSS_XML_DOM_EXCEPTION	Invalid argument type, expected string: value	Cannot set the attribute value with the specified value.

Syntax

```

1 //Add additional code
2 ...
3 var attrValue = attr.value;
4 ...
5 //Add additional code

```

xml.escape(options)



Note: The content in this help topic pertains to SuiteScript 2.0.

Method Description	Prepares a string for use in XML by escaping XML markup, such as angle brackets, quotation marks, and ampersands.
Returns	string
Supported Script Types	<p>All script types For more information, see the help topic SuiteScript 2.0 Script Types.</p>
Governance	None
Module	N/xml Module

Since	2015.2
--------------	--------

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.xmlText	string	Required	String being escaped.	2015.2

Syntax

```

1 //Add additional code
2 ...
3 var xmlEscapedDocument = xml.escape({
4     xmlText : xmlFileContent
5 });
6 ...
7 //Add additional code

```

xml.validate(options)

Note: The content in this help topic pertains to SuiteScript 2.0.
--

Method Description	Validates an XML document against an XML Schema (XSD).
	Important: This method only validates XML Schema (XSD); validation of other XML schema languages is not supported.
	The XML document must be passed as an xml.Document object. The location of the source XML Document does not matter; the validation is performed with the Document object stored in memory. The XSD must be stored in the File Cabinet.
Returns	void
Supported Script Types	All server-side script types For more information, see the help topic SuiteScript 2.0 Script Types .
Governance	None
Module	N/xml Module
Since	2015.2

Parameters

Note: The options parameter is a JavaScript object.
--

Parameter	Type	Required / Optional	Description	Since
options.xml	xml.Document	Required	The xml.Document object to validate.	2015.2

Parameter	Type	Required / Optional	Description	Since
options.xsdFilePathOrId	number string	Required	The file ID or path to the XSD in the File Cabinet to validate the XML document against.	2015.2
options.importFolderPathOrId	number string	Optional	The folder ID or path to a folder in the File Cabinet containing additional XSD schemas which are imported by the parent XSD.	2015.2

Errors

Error Code	Thrown If
SSS_XML_DOES_NOT_CONFORM_TO_SCHEMA	The provided XML is invalid for the provided schema.
SSS_INVALID_XML_SCHEMA_OR_DEPENDENCY	Schema is an incorrectly structured XSD or the dependent schema cannot be found.

Syntax

```

1 //Add additional code
2 ...
3 xml.validate({
4   xml : xmlDocument,
5   xsdFilePathOrId : 'SuiteScripts/schema_parent.xsd',
6   importFolderPathOrId : 'SuiteScripts/'
7 });
8 ...
9 //Add additional code

```

xml.NodeType

Note: The content in this help topic pertains to SuiteScript 2.0.

Enum Description	Enumeration that holds the string values for the supported node types. The Node.nodeType property is defined by one of the values in this enum. Use this enum to determine the type of a node in an XML document.
	Note: Enum values are constants and therefore read-only.
Supported Script Types	All script types For more information, see the help topic SuiteScript 2.0 Script Types .
Module	N/xml Module
Since	2015.2

Values

■ ATTRIBUTE_NODE	■ DOCUMENT_NODE	■ ENTITY_REFERENCE_NODE
------------------	-----------------	-------------------------

■ CDATA_SECTION_NODE	■ DOCUMENT_TYPE_NODE	■ NOTATION_NODE
■ COMMENT_NODE	■ ELEMENT_NODE	■ PROCESSING_INSTRUCTION_NODE
■ DOCUMENT_FRAGMENT_NODE	■ ENTITY_NODE	■ TEXT_NODE

Syntax

```
1 //Add additional code
2 ...
3 var DocType = xmlDocument.nodeType; \\ returns DOCUMENT_NODE
4 ...
5 //Add additional code
```