

QRULES COMMAND OVERVIEW

PRODUCT: qRules v8.1
LAST UPDATED: December 2, 2016

This document includes an overview of all commands included in qRules.

If you are looking for a specific command, you can refer to the **Command Index** at the end of this guide, where you will find all commands listed alphabetically.

TABLE OF CONTENTS

Table of contents	1
Resources	2
Command overview	3
Data Connection	3
Date	6
DBXL	9
File	13
Financial	15
Form	20
Math	29
Misc or Email	35
Node	37
SharePoint	44
String	54
Command compatibility matrix	60
Command index	63
Support	65
Contact Information	65



RESOURCES

If you have not done so already, we suggest your qRules journey begin with the **qRules Installation Guide**, included alongside the qRules installer.

The **qRules Quick Start** document will walk you through injecting an InfoPath form and setting up one command. This will help illustrate the basic usage of qRules.

You could then:

- Check out Qdabra's qRules training modules and videos, which are available by clicking [here](#).
- Try the **qRulesForm.xsn** to see command samples. In default installations, this form is included in the following folder: *C:\Program Files\Qdabra Software\Qdabra Rules Library*. To use this form, navigate to this folder, right-click on qRulesForm.xsn and select **Design**. Make sure that you select the default view, called **Main**, and click **Preview**.
- Refer to the **qRules User Guide** for various hints and tips as you work with Qdabra qRules.
- **qRules Full Trust Form Installation Guide**: Explains how to install the "Qdabra InfoPath to SharePoint List Tool". In default installations, you'll find this in *C:\Program Files\Qdabra Software\Qdabra Rules Library\Forms\Installable Forms*.

Please refer to the qRules product page on Qdabra.com for an updated list of all available documentation.

INJECTOR OPTIONS

The qRules injector has a number of options available. Generally, use the version of InfoPath that matches the compatibility of your template, and, unless needing rich client only commands, the Sandbox option. Only a few commands require using the Admin Deploy injection option – even if you intend to admin deploy your form.

Those are:

- SubmitAsUser
- QueryData, if the **ssapid** parameter is used
- RenderForm, if used in the browser
- SubmitToSharePointList, if the **ssapid** parameter is used

Please review the [Command Compatability Matrix](#) at the end of this guide for additional command requirements.



COMMAND OVERVIEW

The general operation of qRules in a form template is as follows:

1. The form designer adds a rule to the form (on a button or a field) that sets the value of the QdabraRules (secondary data source) **Command** node. This value can be set either as a direct text value or a formula.
2. The compiled qRules code attempts to execute the command and then clears the command field so it is ready for the next command.
3. Errors are returned to the **Error** node. The **Success** node is set to true if the Command execution succeeds. Any returned results are returned the **Results** node – you can also use the **/resultxpath** and **/resultdestds** to return results to a specific form field.

Commands may or may not include parameters. The syntax for a parameter is:

`/parametername=parametervalue`

Parameters are separated by a space from the command and from each other. In the command below, the spaces have been widened to make them more visually apparent:

`DateAdd /xpath=/my:myFields/my:StartingDate /weekdays=5`

Data Connection

ChangeConnectionUrl

Changes the URL for a data connection (web service, XML file, SharePoint List, etc.).

- **Parameters:**
 - **dsname:** Name of data connection where URL should be changed
 - **url:** New URL for the data connection
 - **listguid** (optional): If the form is InfoPath 2010 or greater, this command can be used to change SharePoint List data connections. The **/listguid** parameter should be the GUID for the list.
- **Examples:**
 - `ChangeConnectionUrl /dsname=GetMyInfo /url=http://servername/QdabraWebService/ADUserInfo.aspx`
 - `concat("ChangeConnectionUrl /dsname=", fieldWithName, " /url=", fieldWithUrl)`
 - `ChangeConnectionURL /dsname=TestList /url=http://sharepoint/site/subsite/list /listguid={b01a63f1-e6ff-4067-a397-8a1f62958767}`
- **Additional Observations:**
 - This command does not execute the data connection but merely changes the URL. You'll need a separate action in your rule to query using the data connection.
 - This command works for web service connections, file query connections (xml), SharePoint List connections in InfoPath 2010 or greater, and file submit connections (SharePoint Library).

ChangeSubmitUrl



Modifies the URL in a SharePoint submit data connection so that forms submit to a different library by specifying /url. If the /url parameter is not specified (and the form is published to a SharePoint form library), it uses the library the form is opened from.

- **Parameters:**
 - **submit:** SharePoint Library Submit Adapter
 - **url** (optional): Path to the SharePoint library to change the submit connection to
- **Examples:**
 - *ChangeSubmitUrl /submit=ShPSubmit*
 - *ChangeSubmitUrl /submit=ShPSubmit /url=http://servername/testlibrary*
 - *concat("ChangeSubmitUrl /submit=ShPSubmit /url=", fieldWithUrl)*
- **Additional Observations:**
 - If the /url parameter is not specified for a filler form that is created from a library other than the library it is submitted to, the command will use the URL of the library with the template – that is, the URL is returned from the href attribute of the form XML. This means that this scenario won't work for a filler form unless you specify the /url.
 - This command does not include the actual submit action. You'll need a separate action in your rule to submit using the data connection.

ExecuteSqlQuery

This command executes a SQL query. The form designer needs to create a SQL Data connection, and pass in the entire command. To use this command in the browser, the data connection needs to be converted to a UDC and the form must be admin-deployed.

Note: This command poses a high risk if used with user input. A user could, for example, use:
1; Drop table foo
 which could result in the table **foo** getting deleted.

- **Parameters:**
 - **dsname:** the name of the SQL DB connection.
 - **sql:** the string of SQL you wish to execute.
 - **timeout** (optional): timeout length, in seconds, for this data connection
- **Examples:**
 - *ExecuteSqlQuery /dsname=SQLConnect /sql=SELECT id, name, author, description FROM Database.dbo.Table_1*
 - *concat("ExecuteSqlQuery /dsname=SQLConnect /sql=SELECT id, name, author, description FROM Table_1 where id=", field1)*
 - *ExecuteSqlQuery /dsname=SQLConnect /sql=EXEC dbo.GetUsers @department='HR' /timeout=120*
- **Additional Observations:**
 - When creating the initial data connection, use a SQL query that will generate a result set. If the form is going to execute a SQL query that returns a result set (with, for example, 4 columns returned) set up the data connection with similar SQL so that the schema is correct.

GetConnectionUrl

Returns the current URL of the specified data connection.

- **Parameters:**



- **dsname**: The name of the data connection
- **Examples:**
 - *GetConnectionUrl /dsname=SharePoint Library Submit*

IsDestReachable

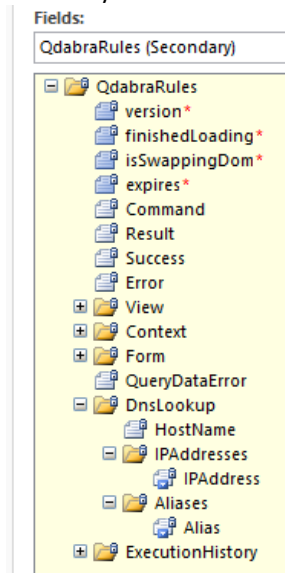
Checks whether the url provided is reachable.

- **Parameters:**
 - **url**: URL to check
- **Examples:**
 - *IsDestReachable /url=http://www.google.com*
- **Additional Observations:**
 - IsDestReachable is not available in the browser.

LookupDns

Allows looking up DNS information.

- **Parameters:**
 - **type** (optional): The type of lookup to perform. Should be one of: GetHostEntry, GetHostByIpAddress, GetHostIpAddresses, GetLocalHostName. Defaults to GetHostEntry if unspecified.
 - **value** (optional): The value (hostname or IP address) to look up. Required for all lookup types except GetLocalHostName.
- **Examples:**
 - *LookupDns /type=GetHostEntry /value=qdabra.com*
 - *LookupDns /type=GetHostByIpAddress /value=184.169.176.91*
- **Additional Observations:**
 - This command is not compatible with Office 365.
 - In SharePoint 2013, this command requires admin deploy.
 - In SharePoint 2010, forms using this command can be sandboxed.
 - Refer to the QdabraRules secondary data source for results of the Dns lookup:



ModifySqlQuery

This command will allow you to add a WHERE clause to an existing SQL data connection. It will execute the SQL query after adding the WHERE clause.

Note: This command poses a high risk if used with user input. A user could, for example, use:
1; Drop table foo
 which could result in the table **foo** getting deleted.

- **Parameters:**
 - **dsname:** the name of the SQL DB connection. This SQL data connection cannot contain a WHERE clause already, sorting or joins.
 - **where:** the WHERE clause that will be added to the SQL query.
- **Examples:**
 - `ModifySqlQuery /dsname=SQLConnect /where=where Location='Office A'`
 - `concat("ModifySqlQuery /dsname=SQLConnect /where=where Location='",
fieldWithLocation, "'")`
- **Additional Observations:**
 - When using this command in a SharePoint 2010 browser form, your SQL data connection must be converted to UDCX and the form must be admin deployed. The command won't work in a SharePoint 2010 sandbox form.

QueryData

Executes a data connection – can be used with web service, XML, database and SharePoint Library Submit connections. Can be used to extend the data connection timeout.

- **Parameters:**
 - **dsname:** Name of the data source to query
 - **url** (optional): service or resource URL. Allows overriding the URL for the data connection. It is ignored unless the connection is one of the following types: WebServiceConnection, FileSubmitConnection, FileQueryConnection.
 - **timeout** (optional): timeout length, in seconds, for this data connection
- **Examples:**
 - `QueryData /dsname=EnumerateDocumentTypeNames`
 - `QueryData /dsname=MyXmlFile /url=http://mySharePointServer/Documents/myXmlFile.xml`
 - `concat("QueryData /dsname=MyReallyBigXmlFile /timeout=" fieldWithTimeout)`
- **Additional Observations:**
 - QueryData is especially useful for error handling. When a rule-based query fails, the user gets a disruptive and sometimes confusing dialog box and rules stop running. With QueryData, the error can be caught and handled for.

Date

ConvertTimeZone

Converts the specified datetime into the destination time zone.

- **Parameters:**
 - **datetime:** The date time to convert.
 - **tzresult:** The desired time zone to convert the date time to.



- **tzsource** (optional): The current time zone for the date time. This defaults to local if used in a Client form, or server time in a browser form.
- **format** (optional): The format for the result. It can be standard or custom date time format string.
- **Examples:**
 - *ConvertTimeZone /datetime=2013-05-24T11:15:48 /tzresult=E. South America Standard Time /tzsource=Pacific Standard Time*
- **Additional Observations:**
 - Both the **tzresult** and the **tzsource** parameters must use the .NET time zone IDs. Here's a list: <http://stackoverflow.com/questions/7908343/list-of-timezone-ids-for-use-with-findtimezonebyid-in-c>
 - Here's an XML file that can be used to populate Timezone dropdowns: <http://www.infopathdev.com/files/folders/103971/download.aspx>

DateAdd

Adds to a date or datetime field.

- **Parameters:**
 - **dsname** (optional): only needs to be specified if the XPath is not located in the Main data source.
 - **xpath**: XPath to starting Date or DateTime field.
 - At least one of the following must be specified as a parameter. Each have a positive or negative value:
 - **seconds**
 - **minutes**
 - **hours**
 - **days**
 - **weekdays**
 - **weeks**
 - **months**
 - **years**
 - **dsnamehol** (optional): if using the /weekdays parameter, a secondary data source can be included for holidays to exclude. Use with /xpathhol to indicate a location for excluded holiday dates.
 - **xpathhol** (optional): if using the /weekdays parameter, /xpathhol allows you to include a repeating field or field in a repeating group with holidays that should be excluded in the date math. If the values are in your main data source, /dsnamehol is not needed.
- **Examples:**
 - *DateAdd /xpath=/my:myFields/my:StartingDate /seconds=60*
 - *DateAdd /xpath=/my:myFields/my:StartingDate /weekdays=5*
 - *DateAdd /xpath=/my:myFields/my:StartingDate /weekdays=5 /dsnamehol=MySharePointHolidayList /xpathhol=/dfs:myFields/dfs:dataFields/d:SharePointListItem_RW/d:HolDate*
 - *concat("DateAdd /xpath=/my:myFields/my:StartingDate /hours=" fieldWithHoursToAdd)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

DateDiff

Calculates the difference between two dates.



- **Parameters:**
 - **date1:** the start date
 - **date2:** the date to subtract from date1
 - **component:** desired portion of the date difference (days | weekdays | hours | minutes | seconds | milliseconds | totaldays | totalhours | totalminutes | totalseconds | totalmilliseconds)
 - **dsnamehol** (optional): if using /component=weekdays, a secondary data source can be included for holidays to exclude. Use with /xpathhol to indicate a location for excluded holiday dates.
 - **xpathhol** (optional): if using /component=weekdays, /xpathhol allows you to include a repeating field or field in a repeating group with holidays that should be excluded in the date math. If the values are in your main data source, /dsnamehol is not needed.
- **Examples:**
 - *DateDiff /date1=2000-01-01T13:00:01 /date2=2000-01-01T13:00:00 /component=totaldays*
 - *DateDiff /date1=2012-10-15 /date2=2012-10-12 /component=days*
 - *concat("DateDiff /date1=", fieldWithStartDate, " /date2=", fieldWithDateToSubtract, " /component=weekdays /dsnamehol=MySharePointHolidayList /xpathhol=/dfs:myFields/dfs:dataFields/d:SharePointListItem_RW/d:HolDate")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

FormatDate

Allows formatting the dates to show just the year, month, or day.

- **Parameters:**
 - **date:** Date to format
 - **format** (optional): Format code for date. Refer to [this article](#) for a list of custom date format codes. Standard date form codes can be found [here](#). If no format is provided, short date pattern is returned
- **Examples:**
 - *FormatDate /date=3/1*
 - *FormatDate /date=2010-03-09T00:00:00 /format=dddd MMMM dd yyyy*
 - *concat("FormatDate /date=", date, " /format=", format)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GetDayOfYear

Returns an integer value indicating the day number in the year for a date value. If no date parameter is provided, uses today's date.

- **Parameters:**
 - **date(optional):** the date value to return the day of year for, defaults today's date.
- **Examples:**
 - *GetDayOfYear*
 - *GetDayOfYear /date=2011-05-04*
 - *concat("GetDayOfYear /date=", datefield)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.



GetFullDateTime

This command returns milliseconds so that you can use date time as a unique ID. Supports optional format.

- **Parameters:**
 - **format** (optional): Format code for date. Refer to [this article](#) for a list of custom date format codes. Standard date form codes can be found [here](#).
- **Examples:**
 - *GetFullDateTime*
 - *GetFullDateTime /format=o*
 - *concat("GetFullDateTime /format=" fieldWithFormat)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GetTimeZone

Obtains the user's current timezone.

- **Parameters:**
 - none
- **Examples:**
 - *GetTimezone*
- **Additional Observations:**
 - In a browser form, this command obtains the server's time zone.

GetWeekDay

Gets the date value for the specified day of the week.

- **Parameters:**
 - **date** (optional): Date for the week; defaults to today's date if not specified
 - **day**: Day to get week for; takes the following formats:
 - Monday or Mo
 - Tuesday or Tu
 - Wednesday or We
 - Thursday or Th
 - Friday or Fr
 - Saturday or Sa
 - Sunday or Su
- **Examples:**
 - *GetWeekDay /day=Sunday*
 - *GetWeekDay /date=03/10/2010 /day=Mo*
 - *concat("GetWeekDay /date=", WeekDate, " /day=", WeekDay)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

DBXL

The following commands all require Qdabra's web service suite, [DBXL](#). Some commands require specific versions of DBXL – which is noted in the command description.



AssignDocument

Creates a QdFlow document, which provides assignment and status workflow for a specific document. Note that this command requires Qdabra's DBXL v2.4. The use of the /email and the /ccEmail parameters require DBXL version 2.5 or greater.

- **Parameters:**
 - **submit:** Name of the *query* (receive) data connection to DBXL (please refer to the **qRules User Guide** for details)
 - **docid:** DocID of the target document
 - **comment:** QdFlow allows comments to be included with the assignment (optional)
 - **status:** Status that will be bestowed upon the target document (optional). If Status is not declared, it will remain blank.
 - **alias:** User alias of the user being assigned to the target document
 - **name:** Name of the user being assigned to the target document (optional)
 - **email:** Email of the user being assigned to the target document (optional)
 - **cc:** Semi-colon separated aliases of users that will be Cc'd in any email notifications as a result of this assignment (optional)
 - **ccEmail:** Semi-colon separated emails of users that will be Cc'd in any email notifications as a result of this assignment (optional, but if included requires the same number of aliases in the cc parameter)
- **Examples:**
 - `concat("AssignDocument /submit=", xpathstosubmit, " /docid=", xpathstodocid, " /alias=", xpathstoalias)`
 - `AssignDocument /submit=SubmitDocument /docid=13 /comment=These are the comments. /status=Submitted /alias=jsmith /name=John Smith /cc=mjohnson;ckent`
 - `AssignDocument /submit=SubmitDocument /docid=13 /comment=These are the comments. /status=Submitted /alias=jsmith /name=John Smith /email=jsmith@company.com /cc=mjohnson;ckent /ccEmail=mjohnson@company.com;ckent@company.com`
- **Additional Observations:**
 - This command does not validate the values provided. This means that you need to make sure that the values for HostDocID, AssignedToAlias, AssignedToName and any CCAliases are valid.
 - All parameters are literal strings. If you need to specify XPath values for name, author and description, make sure to use a concat operation, as shown in the example above.

RemoveDbxIPi

Removes the DBXL Processing Instructions from an xml document. This command takes no parameters.

- **Parameters:**
 - none
- **Examples:**
 - `RemoveDxIPi`
- **Additional Observations:**
 - Results node indicates *Success* or *Fail*.

ResetDbxlVersion



Removes the version information from an xml document that contains DBXL processing instructions. Useful to allow forms to overwrite newer versions if that is the desired behavior.

- **Parameters:**
 - **version** (optional): name of the processing instruction to remove. If version is not supplied, the command removes the version from the pi.
 - **dsname** (optional): secondary data source name
- **Examples:**
 - *ResetDbxlVersion*
 - *ResetDbxlVersion /version=5*
 - *concat("ResetDbxlVersion /version=", fieldWithVersion, " /dsname=SecondaryDataSouce")*

SaveToDbxl

Saves a file, a set of files, an image or a set of images to a DBXL DocType called QdFile (please refer to the **qRules User Guide** for details).

- **Parameters:**
 - **submit**: Name of the *query* (receive) data connection to DBXL (please refer to the **qRules User Guide** for details).
 - **dsname**: data source (optional): only needs to be specified if the data source is not the Main data source.
 - **xpath**: XPath to file/image attachment
 - **qdimage** (optional): can be *yes* or *no* – defaults to *yes*. Create a QdImage document for a saved picture.
- **Examples:**
 - *SaveToDbxl /submit=DBXLSubmit /xpath=/my:myFields/my:File*
 - *SaveToDbxl /submit=DBXLSubmit /xpath=/my:myFields/my:Images/my:Image*
- **Additional Observations:**
 - Source field must have **qRulesLink** and **qRulesFilename** attributes (please refer to the **qRules User Guide** for details).
 - If uploading more than one file/image, the source field must be repeating (not a field in a repeating group (please refer to the **qRules User Guide** for details)
 - qRules will perform no action if the user calls the rule with no image/file attached. When designing your InfoPath solution, you can design your form such that the rule only executes when the image/file attachment is not empty (blank).

SubmitToDbxl

Submits the document to Qdabra DBXL.

- **Parameters:**
 - **submit**: name of the *query* (receive) data connection to DBXL (please refer to the **qRules User Guide** for details)
 - **timeout** (optional): allows you to enter the timeout in secs. This can be used if you have large documents to submit to DBXL.
 - **name** (optional): form name
 - **author** (optional): form author
 - **desc** (optional): form description
 - **dsname** (optional): allows you to submit a datasource other than the main data source to DBXL. This can be used when you have child doc types and have added their XML as a 2ds



- **doctype** (optional): allows you to explicitly specify the doc type to submit to. This is necessary when the template is going to be accessed from somewhere other than DBXL (like SharePoint) or when you're submitting a data source other than the main data source.
- **docid** (optional): DocID of the document you are submitting. If not provided, DBXL automatically assigns one.
- **key** (optional): If submitting via the SubmitDocumentWithKey web method, the key parameter must be provided.
- **user** (optional): If submitting via the SubmitDocumentAsUser web method, the user parameter must be provided.
- **doctypexpath** (optional): Relative XPath to submit doc type value; overrides /doctype param.
- **xpath** (optional): XPath to XML-tree node(s) to submit, if you are submitting only a portion of the xml. If submitting a sub-tree, the following parameters apply:
 - **docidxpath**: Relative XPath to submit DocID value (optional)
 - **keyxpath**: Relative XPath to submit key value (optional)
 - **namexpath**: Relative XPath to Name value; overrides /name param (optional)
 - **authorxpath**: Relative XPath to Author value; overrides /author param (optional)
 - **descxpath**: Relative XPath to Description value; overrides /desc param (optional)
 - **resultdocidxpath**: Relative XPath to store DocID value returned from successful submit (optional)
 - **resultrefidxpath**: Relative XPath to store RefID value returned from successful submit (optional)
- **Examples:**
 - *SubmitToDbxl /submit=SubmitDocument /name=Name /author=Author /desc=Description*
 - *SubmitToDbxl /submit=SubmitDocument /doctype=DocTypeName /name=Name /author=Author /desc=Description*
 - *SubmitToDbxl /submit=DBXLSubmit /namexpath=/my:myFields/my:HistorySection/my:Assignment/my:AssignedToAlias/@my:AssignedToName /descxpath=/my:myFields/my:HistorySection/my:Assignment/my:Comments*
- **Additional Observations:**
 - Cannot use concat if parameters are rich text. It will fail with error "Invalid command specified"
 - You may specify the /doctype parameter for more robust retrieval of input parameters.
 - For assistance on using this command to submit an xml sub-tree to DBXL, please contact Qdabra Software.
 - This command does not validate the XML that's being submitted. Therefore, if you'd like to ensure the XML does not contain errors, please execute the command GetErrorCount to see if there are validation errors. You can then execute SubmitToDBXL IF GetErrorCount returns aa 0 to the Result field.
 - This command also works with the DbxlDocumentWithUserKeyService.

SwapDomWithDocument

After retrieving documents by passing in the DBXL document ID in the input parameters, this command replaces the main DOM with the retrieved XML.

- **Parameters:**
 - **getdocument**: DBXL GetDocument data connection name
 - **docid**: document ID to replace the DOM with



- **version** (optional): Added in qRules v5.1. This allows querying a previous version of a form stored in DBXL. If using this parameter, use GetDocumentVersion instead of GetDocument in your data connection.
- **upgradedocument** (optional): Boolean parameter that attempts to apply the template's built in upgrade to the incoming document.
- **Examples:**
 - *SwapDomWithDocument /getdocument=GetDocument /docid=123*
 - *concat("SwapDomWithDocument /getdocument=GetDocument /docid=", DocID)*
 - *SwapDomWithDocument /getdocument=GetDocumentVersion /docid=8645 /version=2*
- **Additional Observations:**
 - This command also works with the DbxlDocumentWithUserKeyService.

File

AlterImage

Modifies an image attached to a picture control in the form. /image or /imagexpath must be specified.

- **Parameters:**
 - **image**: The base64 value for an image
 - **imagexpath**: The XPath for the image to modify
 - **imageids** (optional): The data source for the image to modify (when using /imagexpath)
 - **width** (optional): The new height for the image
 - **height** (optional): The new width for the image
 - **maintainproportions** (optional): Boolean. This parameter determines whether to maintain the proportions of the original image
 - **quality** (optional): (number > 0 and <= 100) the quality for the modified image (only applies to certain formats)
 - **format** (optional): The new format for the image
- **Examples:**
 - *AlterImage /imagexpath=/my:myFields/my:field1 /width=50 /maintainproportions=true*
 - *AlterImage /imagexpath=/my:myFields/my:field1 /width=200 /maintainproportions=true /quality=50*
- **Additional Observations:**
 - Note: picture controls are not available in 2007 browser forms.
 - The /format parameter can have the following values: jpg, jpeg, gif, png, bmp, tif and tiff. Variants like jpg/jpeg are equivalent. One way to verify the format change is to display the first four characters of the field value in an expression box. The expected prefix for each file type would be:
 - jpg – iVBO
 - gif – R0IG
 - png – iVBO
 - tif – Sukq
 - bmp – Qk1* (the * can be any character)

CreateDirectory

Creates a local directory. Full trust required. Not browser compatible.

- **Parameters:**



- **path**: Path of the directory to create.
- **Examples:**
 - *CreateDirectory /path=C:\test*
 - *concat("CreateDirectory /path=", fieldWithPath)*

DecodeBase64

Decodes text based files attached to a form.

- **Parameters:**
 - **sourceds** (optional): only needs to be specified if the XPath to attachment is not located in the Main data source.
 - **sourcepath**: XPath of the file attachment field
 - **destds** (optional): data source where the decoded XML should be returned (if using /asxml=true and destpath is in a secondary data connection).
 - **destpath** (optional): XPath of the location where the decoded XML should be returned (if using /asxml=true – also include /destds if the location is in a secondary data source).
 - **asxml** (optional): convert the decoded string to xml and return to the specified /destpath. Can be true or false, defaults to true.
 - **excludesourceroot** (optional): when treating the decoded string as xml, inserts everything except the XML root into the destination location. Can be true or false, defaults to true.
- **Examples:**
 - *DecodeBase64 /sourcepath=/my:myFields/my:field1 /destpath=/my:myFields/my:group /asxml=true*
 - *DecodeBase64 /sourcepath=/my:myFields/my:field1 /asxml=false*
 - *DecodeBase64 /sourcepath=/my:myFields/my:field1 /destpath=/my:myFields /asxml=true*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

DeleteFile

Deletes a local file. Full trust required. Not browser compatible.

- **Parameters:**
 - **path**: path of the file to delete.
- **Examples:**
 - *DeleteFile /path=C:\text.xml*
 - *concat("DeleteFile /path=", fieldWithPathToFileToDelete)*

GetAttachmentAttribute

Allows you the ability to obtain attributes from an attachment included in an InfoPath form in a File Control.

- **Parameters:**
 - **dsname** (optional): only needs to be specified if the XPath to attachment is not located in the Main data source.
 - **xpath**: XPath of the file attachment field
 - **attr**: specific attribute value to obtain, such as Name, Ext, Size or File.
- **Examples:**
 - *GetAttachmentAttribute /xpath=/my:myFields/my:FileAttachment /attr=Name*
- **Additional Observations:**



- Returns result to the QdabraRules data source Result node.

ListDirectory

This command takes a path to a directory, returns results to a field in a repeating node. Note that /parent must point to a non-repeating field and /child to a repeating field, relative to the parent. Full trust required. Not browser compatible.

- **Parameters:**
 - **path:** Path to the directory to list.
 - **mode** (optional): *file* OR *directory* to list files or directories. Default is *file*.
 - **dsname** (optional): Data source where the results should be deposited. Defaults to the main data source.
 - **parent:** Parent node where the results should be deposited.
 - **child:** Name of the child node used to list results.
- **Examples:**
 - `ListDirectory /path=C:\Data /parent=/my:myFields/my:Parent /child=my:child`
 - `concat("ListDirectory /path=", fieldWithPath, " /parent=/my:myFields/my:Parent /child=my:child")`

SetAutoUpdate

SetAutoUpdate allows you to control whether the views in your form are automatically synchronized with the data contained in the underlying XML. This can improve form performance, since you may want to perform a large number of updates before refreshing the view.

- **Parameters:**
 - **enabled:** true|false. If **true**, Auto Update is enabled - if **false**, disabled. Not required if supplying a value for /force.
 - **force:** true|false. If **true**, Force Update occurs. Not required if supplying a value for /enabled.
- **Examples:**
 - `SetAutoUpdate /enabled=true`
 - `SetAutoUpdate /enabled=false`
 - `SetAutoUpdate /force=true`
- **Additional Observations:**
 - This command is not available in Browser forms.
 - For additional information, please refer to [this page](#) for InfoPath 2007, [this page](#) for 2010, and [this page](#) for 2013.

Financial

The following commands are qRules implementations of the Microsoft.VisualBasic.Financial Methods. More information can be found [here](#).

Ddb

Returns the depreciation of an asset for a specific time period using the double-declining balance method or some other method you specify.

- **Parameters:**
 - **cost:** initial cost of the asset



- **salvage**: value of the asset at the end of its useful life
- **life**: length of useful life of the asset
- **period**: period for which asset depreciation is calculated
- **factor** (optional): the rate at which the balance declines. Defaults to 2 (double-declining method).
- **Examples:**
 - `Ddb /cost=5000 /salvage=5 /life=3 /period=1`
 - `Ddb /cost=5000 /salvage=5 /life=3 /period=1 /factor=3`
 - `concat("Ddb /cost=", fieldWithCost, " /salvage=", fieldWithSalvage, " /life=3 /period=1 /factor=3")`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Fv

Returns the future value of an annuity based on periodic, fixed payments and a fixed interest rate.

- **Parameters:**
 - **rate**: interest rate per period
 - **nper**: total number of payment periods in the annuity
 - **pmt**: payment to be made each period
 - **pv** (optional): present value (or lump sum) of a series of future payments. Defaults to 0 if not provided.
 - **due** (optional): can be *end*, for End of Period, or *beg*, for Beginning of Period. Defaults to end.
- **Examples:**
 - `Fv /rate=.3 /nper=12 /pmt=120`
 - `Fv /rate=.3 /nper=12 /pmt=120 /pv=-15000`
 - `Fv /rate=.3 /nper=12 /pmt=120 /pv=-15000 /due=beg`
 - `concat("Fv /rate=", fieldWithRate, " /nper=12 /pmt=120 /pv=-15000")`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Ipmt

Returns the interest payment for a given period of an annuity based on periodic, fixed payments and a fixed interest rate.

- **Parameters:**
 - **rate**: interest rate per period
 - **per**: payment period in the range 1 through /nper
 - **nper**: total number of payment periods in the annuity
 - **pv**: present value (or lump sum) of a series of future payments
 - **fv** (optional): future value, or cash balance you want after the final payment is made. Defaults to zero.
 - **due** (optional): can be *end*, for End of Period, or *beg*, for Beginning of Period. Defaults to end.
- **Examples:**
 - `Ipmt /rate=.3 /per=1 /nper=12 /pv=-250`
 - `Ipmt /rate=.3 /per=1 /nper=12 /pv=-250 /fv=100 /due=end`
 - `concat("Ipmt /rate=.3 /per=1 /nper=", fieldWithValue, " /pv=-250 /fv=100 /due=end")`



- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - The /per parameter must be 1 or greater and less than the value of the /nper parameter.

Irr

Returns the internal rate of return for a series of periodic cash flows (payments and receipts). The [DelimitedList](#) command could be used to obtain the comma delimited list required for the /value parameter.

- **Parameters:**
 - **values:** comma delimited list of cash flow values – must contain at least one negative value and at least one positive value
 - **guess** (optional): estimated rate. Default is 0.1 (10 percent).
- **Examples:**
 - *Irr /values=3,4,5,-6*
 - *Irr /values=3,4,5,-6 /guess=.2*
 - *concat("Irr /values=", [fieldWithCommaDelimitedValuesList](#))*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - The payment and receipt values in the /values parameter need to be in the correct sequence.
 - The function used by the command is iterative. It starts with the value of the /guess parameter cycles through the calculation until the result is accurate within 0.00001 percent – if a result isn't found after 20 tries, it fails (per [MSDN](#)).

Mirr

Returns the modified internal rate of return for a series of periodic cash flows (payments and receipts). The [DelimitedList](#) command could be used to obtain the comma delimited list required for the /value parameter.

- **Parameters:**
 - **values:** comma delimited list of cash flow values – must contain at least one negative value and at least one positive value
 - **frate:** finance rate – the interest rate paid as the cost of financing
 - **rrate:** reinvest rate – the interest rate received on gains from cash reinvestment
- **Examples:**
 - *Mirr /values=3,4,5,-6 /frate=.2 /rrate=.3*
 - *concat("Mirr /values=", [fieldWithCommaDelimitedValuesList](#), " /frate=.2 /rrate=.3")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - The payment and receipt values in the /values parameter need to be in the correct sequence.

Nper

Returns the number of periods for an annuity based on periodic fixed payments and a fixed interest rate.

- **Parameters:**
 - **rate:** interest rate per period (i.e., APR divided by 12, if making monthly payments)



- **pmt**: payment to be made each period
- **pv**: present value (or lump sum) of a series of future payments
- **fv** (optional): future value, or cash balance you want after the final payment is made. Defaults to zero.
- **due** (optional): can be *end*, for End of Period, or *beg*, for Beginning of Period. Defaults to end.
- **Examples:**
 - `Nper /rate=.3 /pmt=120 /pv=-150`
 - `Nper /rate=.3 /pmt=120 /pv=-150 /fv=100 /due=beg`
 - `concat("Nper /rate=.3 /pmt=", fieldWithPayment, " /pv=-150 /fv=100 /due=beg")`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Npv

Returns the net present value of an investment based on a series of periodic cash flows (payments and receipts) and a discount rate. The [DelimitedList](#) command could be used to obtain the comma delimited list required for the `/value` parameter.

- **Parameters:**
 - **rate**: discount rate over the length of the period
 - **values**: comma delimited list of cash flow values – must contain at least one negative value and at least one positive value
- **Examples:**
 - `Npv /rate=.3 /values=3,4,-6`
 - `concat("Npv /rate=", fieldWithRate, " /values=", fieldWithCommaDelimitedValuesList)`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - The payment and receipt values in the `/values` parameter need to be in the correct sequence.

Pmt

Returns the payment for an annuity based on periodic, fixed payments and a fixed interest rate.

- **Parameters:**
 - **rate**: interest rate per period (i.e., APR divided by 12, if making monthly payments)
 - **nper**: total number of payment periods in the annuity
 - **pv**: present value (or lump sum) of a series of future payments
 - **fv** (optional): future value, or cash balance you want after the final payment is made. Defaults to zero.
 - **due** (optional): can be *end*, for End of Period, or *beg*, for Beginning of Period. Defaults to end.
- **Examples:**
 - `Pmt /rate=.3 /nper=12 /pv=15`
 - `Pmt /rate=.3 /nper=12 /pv=15 /due=beg`
 - `concat("Pmt /rate=.3 /nper=", fieldWithNumberOfPayments, " /pv=15 /due=beg")`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Ppmt



Returns the principal payment for a given period of an annuity based on periodic fixed payments and a fixed interest rate.

- **Parameters:**
 - **rate:** interest rate per period (i.e., APR divided by 12, if making monthly payments)
 - **per:** payment period in the range 1 through /nper
 - **nper:** total number of payment periods in the annuity
 - **pv:** present value (or lump sum) of a series of future payments
 - **fv** (optional): future value, or cash balance you want after the final payment is made. Defaults to zero.
 - **due** (optional): can be *end*, for End of Period, or *beg*, for Beginning of Period. Defaults to end.
- **Examples:**
 - `Ppmt /rate=.3 /per=1 /nper=12 /pv=15`
 - `Ppmt /rate=.3 /per=1 /nper=12 /pv=15 /due=beg`
 - `concat("Ppmt /rate=.3 /per=" fieldWithPaymentPeriod, " /nper=12 /pv=15")`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - The /per parameter must be 1 or greater and less than the value of the /nper parameter.

Pv

Returns the present value of an annuity based on periodic, fixed payments and a fixed interest rate.

- **Parameters:**
 - **rate:** interest rate per period (i.e., APR divided by 12, if making monthly payments)
 - **nper:** total number of payment periods in the annuity
 - **pmt:** payment to be made each period
 - **fv** (optional): future value, or cash balance you want after the final payment is made. Defaults to zero.
 - **due** (optional): can be *end*, for End of Period, or *beg*, for Beginning of Period. Defaults to end.
- **Examples:**
 - `Pv /rate=.3 /nper=12 /pmt=120`
 - `Pv /rate=.3 /nper=12 /pmt=120 /fv=15 /due=beg`
 - `concat("Pv /rate=.3 /nper=12 /pmt=", fieldWithPayment)`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Rate

Returns the interest rate per period for an annuity.

- **Parameters:**
 - **nper:** total number of payment periods in the annuity
 - **pmt:** payment to be made each period
 - **pv:** present value (or lump sum) of a series of future payments
 - **fv** (optional): future value, or cash balance you want after the final payment is made. Defaults to zero.
 - **due** (optional): can be *end*, for End of Period, or *beg*, for Beginning of Period. Defaults to end.



- **guess** (optional): estimated rate. Default is 0.1 (10 percent).
- **Examples:**
 - *Rate /nper=12 /pmt=120 /pv=-12000*
 - *Rate /nper=12 /pmt=120 /pv=-12000 /fv=15 /due=beg /guess=.5*
 - *concat("Rate /nper=12 /pmt=", fieldWithPayment , " /pv=-12000")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - The function used by the command is iterative. It starts with the value of the */guess* parameter cycles through the calculation until the result is accurate within 0.00001 percent – if a result isn't found after 20 tries, it fails (per [MSDN](#)).

Sln

Returns the straight-line depreciation of an asset for a single period.

- **Parameters:**
 - **cost:** initial cost of the asset
 - **salvage:** value of the asset at the end of its useful life
 - **life:** length of useful life of the asset
- **Examples:**
 - *Sln /cost=1200 /salvage=100 /life=120*
 - *concat("Sln /cost=1200 /salvage=100 /life=", fieldWithLife)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Syd

Returns the sum-of-years digits depreciation of an asset for a specified period.

- **Parameters:**
 - **cost:** initial cost of the asset
 - **salvage:** value of the asset at the end of its useful life
 - **life:** length of useful life of the asset
 - **period:** period for which asset depreciation is calculated
- **Examples:**
 - *Syd /cost=1200 /salvage=100 /life=120 /period=15*
 - *concat("Syd /cost=1200 /salvage=", fieldWithSalvage , " /life=120 /period=15")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Form

ClearErrors

Clears all validation errors in the InfoPath form. This allows you to clear the error table in order to allow submit even if some fields are not valid, for example. This command takes no parameters. Please note, that when the form is reopened, validation errors will be re-added to the error table, and would need to be cleared again if submit is to be allowed.

- **Parameters:**
 - none
- **Examples:**



- *ClearErrors*

ExecuteAction

Executes a view action. This command is not available in browser forms. Please see MSDN for more information:

2007: [http://msdn.microsoft.com/en-us/library/aa942991\(v=office.11\).aspx](http://msdn.microsoft.com/en-us/library/aa942991(v=office.11).aspx)

2010 & 2013: [http://msdn.microsoft.com/en-us/library/aa942991\(v=office.14\).aspx](http://msdn.microsoft.com/en-us/library/aa942991(v=office.14).aspx)

- **Parameters:**
 - **action:** Action to execute (as according to InfoPath documentation)
 - **xmltoedit** (optional): argument for some actions
- **Examples:**
 - *ExecuteAction /action=XOptionalInsert /xmltoedit=group3_19*

ExportView

Uses InfoPath's built-in functionality to render the current view as a file and save it to a local drive.

- **Parameters:**
 - **path:** Location to save the file.
 - **format** (optional): File format. Available options are: pdf, mht, xps. If not specified, it will default to pdf.
- **Examples:**
 - *ExportView /path=D:\Test forms\test.xps /format=xps*
- **Additional Observations:**
 - ExportView is not available for browser forms.
 - This command requires the form to be Full Trust.
 - The path provided must exist. The command does not create a path that doesn't exist and does not tell the user that the path does not exist.
 - The user must have write permission to the path.
 - The filename must be specified, including the file extension.
 - The extension on the specified file name should typically match the format being used. Otherwise, users can have trouble opening the exported files. For example: *ExportView /path=D:\Test forms\test.mht /format=mht* vs. *ExportView /path=D:\Test forms\test.pdf /format=mht*

GetEnvironment

GetEnvironment returns **Mobile**, **Browser**, or **Client**. This command takes no parameters. You can also use with [SetDefaultView](#) to choose which view a form opens with depending on the environment

- **Parameters:**
 - none
- **Examples:**
 - *GetEnvironment*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.



- Sandboxed forms do not return the correct result for the IsMobile property (which qRules uses to determine environment). Thus, an IPFS form that is sandboxed will always return **Browser**, even when opened on a mobile device.

GetErrorCount

Returns the number of errors that exist in the current xml. This command takes no parameters.

- **Parameters:**
 - none
- **Examples:**
 - *GetErrorCount*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GetErrors

Lists the errors in the current form.

- **Parameters:**
 - **format** (optional): Format for the errors returned. Possible values are: message, fieldAndMessage
- **Examples:**
 - *GetErrors /format=fieldAndMessage*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GetFileName

Obtains the current xml filename. This command takes no parameters.

- **Parameters:**
 - none
- **Examples:**
 - *GetFileName*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GetFormProperty

Obtain the value of a form property.

- **Parameters:**
 - **key**: name of the property to retrieve.
- **Examples:**
 - *GetFormProperty /key=CurrentUri*
- **Additional Observations:**
 - Allowed values for key (case sensitive):
 - **CurrentUri** - The uri of the currently open XML document. This value can change if the form is saved to a new location while it is open.
 - **OpenedUri** - Same as CurrentUri, except that it is fixed at the value it had when the document was opened.



- **SharePointListUrl** - The URL of the SharePoint list that the form template belongs to.
- **SharePointServerRootUrl** - The Root URL of the SharePoint farm that the template belongs to.
- **SharePointSiteCollectionUrl** - The URL of the SharePoint site collection that the template belongs to.
- **SharePointSiteUrl** - The URL of the SharePoint site that the template belongs to.
- Note that the last four are not supported in 2007 forms, and can return misleading data in content type browser forms.

GetInputParameter

Returns if the key exists and gets the value back. If the key does not exist, FAIL is returned.

- **Parameters:**
 - **key:** name of parameter for desired value
- **Examples:**
 - *GetInputParameter /key=username*
 - *concat("GetInputParameter /key=" fieldWithParamKey)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GetViewInfo

Provides information about the view that is currently open in the form.

- **Parameters:**
 - **property:** the name of the view property to retrieve. Allowed values are **name** and **caption**
- **Examples:**
 - *GetViewInfo /property=name*
 - *GetViewInfo /property=caption*
- **Additional Observation**
 - For the name property, this command will place the view's internal name value in the Result field. For caption, this command will provide the view's display name.

InsertPi

Insert a processing instruction into the xml file.

- **Parameters:**
 - **name:** name of the processing instruction to insert.
 - **data:** data content of the processing instruction
 - **update** (optional): Update the processing-instruction if it already exists. Can be yes or no. Default is no.
 - **dsname** (optional): Name of the DataSource to insert the processing instruction. Assumes you are using the Main data source if it is not specified.
- **Examples:**
 - *InsertPi /name=mso-infoPathSolution /data=name="urn:schemas-microsoft-com:office:infopath:ShPFormLibB:-myXSD-2011-07-30T13-53-43" solutionVersion="1.0.0.7" productVersion="14.0.0.0" PIVersion="1.0.0.0" href="http://server/library/Forms/template.xsn"*



IsDirty

This command checks the dirty flag in the form, which is used to determine whether changes have been made to the form, and returns *true* or *false*. This command takes no parameters. Like the [SetDirty](#) command, IsDirty is not available in InfoPath Forms Services.

- **Parameters:**
 - none
- **Examples:**
 - *IsDirty*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

IsNew

Identifies whether a form is newly created and has not yet been saved nor submitted. This command takes no parameters and returns either *true* or *false*.

- **Parameters:**
 - none
- **Examples:**
 - *IsNew*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

IsReadOnly

Identifies whether a form is in a read-only state where users will not be able to save the form (i.e. they will have to use Save As instead). Forms can be in a read-only state for a variety of reasons, e.g. a form opened from a SharePoint library but is locked because someone else has it open. This command takes no parameters and returns either *true* or *false*.

- **Parameters:**
 - none
- **Examples:**
 - *IsReadOnly*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

ListInputParameters

Provides a list of the available input parameters in the currently opened form, as a comma-separated list of all of their keys.

- **Parameters:**
 - None
- **Examples:**
 - *ListInputParameters*

OpenForm

Opens an XSN form, using an XML file if provided. Not browser compatible.



- **Parameters:**
 - **path:** Path or URL to XSN to open.
 - **data:** Path or URL to XML to open. If /path is also provided, /data will be opened with that template.
 - **dsname** (optional): Data source for node indicated by /xpath parameter to provide data for template indicated by path.
 - **xpath** (optional): XPath for node with data to be used with template provided by /path parameter.
 - **new** (optional): true|false - defaults to true. Set to false to open existing XML. If false, must include /data parameter.
- **Examples:**
 - *OpenForm /path=http://testserver/site/FormLibrary/template.xsn*
 - *OpenForm /path=C:\Users\ernestom\Desktop\qRules\Test forms\ExpenseReport.xsn /data=http://shp2010-test2/ExpenseReportTest/2012-09-28.xml /new=true*
- **Additional Observations:**
 - The OpenForm command is only available in client forms.
 - Either /path or /data must be specified.

OpenUrlInBrowser

Opens the given URL in the default browser. Not browser compatible. A form that uses this command MUST be configured to run with full trust using the **Security and Trust** category of the **Form Options** dialog box.

- **Parameters:**
 - **url:** the URL that will be opened in the default browser
- **Examples:**
 - *OpenUrlInBrowser /url=http://www.qdabra.com*

Print

Prints the form. Not browser compatible. A form that uses this command MUST be configured to run with full trust using the **Security and Trust** category of the **Form Options** dialog box.

- **Parameters:**
 - **dialog:** Can be yes or no. When dialog is yes, InfoPath shows the user the print dialog to select printer, settings, etc.
- **Examples:**
 - *Print /dialog=no*
 - *Print /dialog=yes*
- **Additional Observations:**
 - The print command will fail if there is an action to close the form following it.

Relink

Used to switch the form template, or the version of the template of current document.

- **Parameters:**
 - **href:** location of an XSN template you want to link to the form
 - **version** (optional): The /version parameter would be needed if the XMLs are being switched over to another XSN template specified in href with a smaller version number than the XSN the documents are currently linked to.



- **Examples:**
 - *Relink /href=http://<yourSharePointSite>/<Library2>/Forms/template.xsn*
 - The example above would assign <Library2>'s template for form in another library.
 - *Relink /href=http://<yourSharePointSite>/<CurentLibruary>/Forms/template.xsn /version=1.0.0.20*
- **Additional Observations:**
 - The InfoPath version number determines whether an XML needs to be upgraded when it is opened (if the version is older than the latest template's version) or whether InfoPath should refuse to open the form at all (if the version is newer than the latest template's version). It will not cause InfoPath to open the form in a previous version of the available template.
 - In case specified version is newer than the current version of specified XSN in library, uses get warning "Created with a new version. InfoPath cannot find the new version. Do you want to open with the older version?"
 - Version number is found as follows
 - InfoPath 2010 Designer: File->Advanced Form Option -> Versioning
 - InfoPath 2007 Designer: Tools -> Form Options -> Versioning

RemovePi

Remove a processing instruction from the xml file.

- **Parameters:**
 - **name:** name of the processing instruction to remove
- **Examples:**
 - *RemovePi /name= mso-infoPathSolution*

RenderForm

Generates an html rendering of the form's views.

- **Parameters:**
 - **views** (optional): List of views to render, separated by &
 - **allviews** (optional): (Boolean) render all views
 - **output** (optional): Output encoding - Choices are raw (unencoded) or base64 (base64 encoded), base64 by default
 - **filename** (optional): Filename for the output file (only applies when using base64 encoding)
- **Examples:**
 - *RenderForm /allviews=true*
 - *RenderForm /views=View 1 /output=base64 /filename=test.htm*
- **Additional Observations:**
 - /allviews cannot be true if /views is specified.
 - As of qRules v4.3, the command is not compatible with MOSS 2007 browser forms. It is compatible with SharePoint 2010 browser forms that are admin deployed. When injecting the form template, select the **Browser > Admin Deploy** option or the **Rich Client** option.
 - The RenderForm command requires full trust. As a result, the form must be admin deployed.
 - View names are case sensitive.
 - The filename parameter needs to include an extension.
 - If you copy the result to a field that's bound to a file picker, it will show the filename you specified, or the default filename if you didn't specify one (rendered.htm).

SaveToPath



Saves the main or a secondary data source's XML to a local file. Full trust required. Not browser compatible.

- **Parameters:**
 - **path:** Path to save to.
 - **dsname** (optional): data source name. Defaults to the main data source.
 - **overwrite** (optional): Overwrite existing file if it exists. Default is false.
 - **preservelocation** (optional): Maintain the document's current file location. If false or unspecified, the document's save location is changed to the new path.
- **Examples:**
 - *SaveToPath /path=C:\test\myxml.xml /overwrite=true*

SetCaption

Sets the window caption. Not browser compatible. When injecting your form, you have to select “client only” to have the necessary code injected in your form by the injector.

- **Parameters:**
 - **caption:** The caption for the form window
- **Examples:**
 - *SetCaption /caption=MeetingNotesForm*

SetDefaultView

This command allows you to switch view on open using [GetInputParameter](#) or [GetEnvironment](#) to choose a different view to set as default.

- **Parameters:**
 - **view:** name of the view to switch to
- **Examples:**
 - *SetDefaultView /view=Browser*

SetDirty

SetDirty allows you to set the dirty flag, which is used to determine whether changes have been made to the form, to either true or false. Not browser compatible.

- **Parameters:**
 - **dirty:** value for the flag (true | false, yes | no, 1 | 0).
- **Examples:**
 - *SetDirty /dirty=0*
- **Additional Observations:**
 - If the parameter /dirty is set to anything other than **1**, **true** or **yes**, the flag is set to **false**.

SetFocus

SetFocus sets the focus to the specified XPath. Not browser compatible.

- **Parameters:**
 - **dsname** (optional): data source name
 - **xpath:** Xpath of the field you want to set the focus
- **Examples:**



- *SetFocus /xpath=my:myFields/my:name*
- **Additional Observations:**
 - Please note that not every field can have a focus set to it. You can select Text Box, Rich Text Box or Date Picker.

SetSave

SetSave gives the form designer access to a variety of previously code-only save options, such as dynamically allowing save, or setting the save location. Not browser compatible. When injecting your form, you have to select “client only” to have the necessary code injected in your form by the injector.

- **Parameters:** All parameters are optional, but the command won’t execute without parameters.
 - **save:** Allow or prevent the Save action (allow | block)
 - **savemsg:** Message to display if Save is blocked (* to clear)
 - **saveloc:** Block Save if location starts with this (* to clear)
 - **saveas:** Allow or prevent the Save As action (allow | block)
 - **saveasmsg:** Message to display if Save As is blocked (* to clear)
 - **saveasloc:** Suggested location in Save As dialog (* to clear)
 - **saveasfile:** Suggested filename in Save As dialog (* to clear)
- **Examples:**
 - *SetSave /save=block /savemsg=Save is disabled /saveas=allow /saveasloc=C:\ /saveasfile=MyForm.xml*
- **Additional Observations:**
 - This command provides you three alternatives:
 - Block all
 - Allow all
 - Block only locations that begin with the value of parameter /saveloc

SetXml

Allows setting the XML content of a node based on a string.

- **Parameters:**
 - **destpath:** The path of the node(s) where the XML should be inserted.
 - **value:** The XML value
 - **destds** (optional): The data source where the XML should be set, if not the main data source
 - **excludesourceroot** (optional): Omit the outer element of the xml value and only insert everything inside it into the destination node(s).
- **Examples:**
 - *SetXml /destpath=/secondaryxml/mygroup/testnode /destds=sample /value=test*
 - *SetXml /value=<my:myFields
xmlns:my="http://schemas.microsoft.com/office/infopath/2003/myXSD/2013-11-04T17:48:52"><my:testnode>this is a test value</my:testnode></my:myFields>
/destpath=/my:myFields /excludesourceroot=true*
- **Additional Observations:**
 - If you use /excludesourceroot=true, qRules has to load up the specified XML in order to remove its outer element, so any namespaces used within the XML fragment have to be defined.

ShowDialog



Generates and shows a dialog box to the form user.

- **Parameters:**
 - **text:** Text to display in the message box
 - **buttons:** Buttons for the message box - can be any one of the following: AbortRetryIgnore, OK, OKCancel, RetryCancel, YesNo, YesNoCancel.
 - **caption** (optional): Text to display in the caption for the message box.
 - **icon** (optional): Icon to display in the message box - can be any one of the following: Error, Information, None, Question, Stop, Warning.
- **Examples:**
 - *ShowDialog /text=Click here to continue. /buttons=OKCancel*
- **Additional Observations:**
 - ShowDialog is not available for browser forms.

ValidateUser

This command will accept a username, a domain and a password and will verify that the credentials are correct. Returns *true* or *false* according to the result of the log in attempt. Not browser compatible. A form that uses this command MUST be configured to run with full trust using the **Security and Trust** category of the **Form Options** dialog box.

- **Parameters:**
 - **username:** the user alias for the user being verified. Optional parameter. Defaults to the current user.
 - **domain:** Domain of the user to validate. Optional parameter.
 - **password:** Password to verify. Required parameter.
- **Examples:**
 - *ValidateUser /password=1234*
 - *ValidateUser /username=ernestom /domain=autonomysystems /password=1234*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Math

Abs

Returns the absolute value for the number provided.

- **Parameters:**
 - **value:** number to return absolute for
- **Examples:**
 - *Abs /value=10.345*
 - *concat("Abs /value=", fieldWithValue)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Acos

Returns angle measured in radians or degrees for the provided cosine – cosine value must be greater than or equal to -1, but less than or equal to 1. Returns NaN if outside of range.



- **Parameters:**
 - **value:** cosine for returned angle
 - **output** (optional): can be *rad* or *deg* to return in radians or degrees – default is radians
- **Examples:**
 - *Acos /value=.25*
 - *Acos /value=.25 /output=deg*
 - *concat("Acos /value=", fieldWithValue, " /output=deg")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Asin

Returns the angle measured in radians or degrees for the provided sine - sine value must be greater than or equal to -1, but less than or equal to 1. Returns NaN if outside of range.

- **Parameters:**
 - **value:** sine for returned angle
 - **output** (optional): can be *rad* or *deg* to return in radians or degrees – default is radians
- **Examples:**
 - *Asin /value=.25*
 - *Asin /value=.25 /output=deg*
 - *concat("Asin /value=", fieldWithValue, " /output=deg")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Atan

Returns the angle measured in radians or degrees for the provided tangent. Returns NaN if outside of range.

- **Parameters:**
 - **value:** tangent for returned angle
 - **output** (optional): can be *rad* or *deg* to return in radians or degrees – default is radians
- **Examples:**
 - *Atan /value=.25*
 - *Atan /value=.25 /output=deg*
 - *concat("Atan /value=", fieldWithValue, " /output=deg")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Atan2

Returns the angle measured in radians or degrees whose tangent is the quotient of the provided values. Returns NaN if outside of range.

- **Parameters:**
 - **value1:** first number for quotient (the y coordinate of a point)
 - **value2:** second number for quotient (the x coordinate of a point)
 - **output** (optional): can be *rad* or *deg* to return in radians or degrees – default is radians
- **Examples:**
 - *Atan2 /value1=.25 /value2=.5*



- *Atan2 /value1=.25 /value2=.5 /output=deg*
- *concat("Atan /value1=", fieldWithValue, " /value2=.5 /output=deg")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

BigMul

Returns the full product of two 32-bit numbers.

- **Parameters:**
 - **value1:** first number to multiply
 - **value2:** second number to multiply
- **Examples:**
 - *BigMul /value1=2147483647 /value2=2147483647*
 - *concat("BigMul /value1=", fieldWithValue, " /value2=2147483647")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Ceiling

Returns smallest integral value that is greater than or equal to the value provided

- **Parameters:**
 - **value:** number to return ceiling for
- **Examples:**
 - *Ceiling /value=1.25*
 - *concat("Ceiling /value=", fieldWithValue)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Cos

Returns cosine for the angle provided. Returns NaN if outside of range.

- **Parameters:**
 - **value:** an angle, measured in radians or degrees - for degrees, include /input=deg
 - **input** (optional): can be *rad* or *deg* to indicate whether /value is radians or degrees – default is radians
- **Examples:**
 - *Cos /value=.25*
 - *Cos /value=45 /input=deg*
 - *concat("Cos /value=", fieldWithValue, " /input=deg")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Cosh

Returns hyperbolic cosine for the angle provided

- **Parameters:**
 - **value:** a hyperbolic angle
- **Examples:**
 - *Cosh /value=.25*



- `concat("Cosh /value=", fieldWithValue)`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

DivRem

Returns the quotient and the remainder as comma separated values.

- **Parameters:**
 - **value1:** the dividend
 - **value2:** the divisor (returns an error if zero)
- **Examples:**
 - `DivRem /value1=15 /value2=12`
 - `concat("DivRem /value1=", fieldWithValue, " /value2=12")`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Exp

Returns e raised to the power specified

- **Parameters:**
 - **value:** power to raise e to
- **Examples:**
 - `Exp /value=9`
 - `concat("Exp /value=", fieldWithValue)`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

FormatNumber

Formats a number into the specified format.

- **Parameters:**
 - **value:** number to apply the format to
 - **format:** the standard or custom format for the result. Can be either standard or custom:
 - <http://msdn.microsoft.com/en-us/library/dwhawy9k.aspx>
 - <http://msdn.microsoft.com/en-us/library/0c899ak8.aspx>
 - **culture** (optional): The desired culture provider for the result. Culture is optional and uses the default culture if not specified. The options for culture are listed here: <http://msdn.microsoft.com/en-us/goglobal/bb896001.aspx>
- **Examples:**
 - `FormatNumber /format=c /value=0.749 /culture=pt-BR`

Floor

Returns the largest integer less than or equal to the value provided

- **Parameters:**
 - **value:** number to return floor for
- **Examples:**
 - `Floor /value=1.25`
 - `concat("Floor /value=", fieldWithValue)`



- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GenerateRandomNumber

Returns a random number. Allows setting of upper or upper and lower bounds for the random number.

- **Parameters:**
 - **max** (optional): Exclusive upper bound
 - **min** (optional): Inclusive lower bound, must have a maximum number to use
- **Examples:**
 - *GenerateRandomNumber*
 - *GenerateRandomNumber /max=100 /min=20*
 - *concat("GenerateRandomNumber /max=10 /min=" fieldWithMinValue)*
 - *GenerateRandomNumber /max=100*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Log

Returns the logarithm of the value provided - if newBase is included, in a specified base.

- **Parameters:**
 - **value**: number to find logarithm for
 - **newBase** (optional): base for the logarithm
- **Examples:**
 - *Log /value=1.25*
 - *Log /value=1.25 /newBase=8*
 - *concat("Log /value=", fieldWithValue, " /newBase=8")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Pow

Returns a specified number raised to the specified power.

- **Parameters:**
 - **value**: the number to be raised
 - **power**: the power to raise the number to
- **Examples:**
 - *Pow /value=1.25 /power=8*
 - *concat("Pow /value=", fieldWithValue, " /power=8")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Round

Returns the value rounded - if scale is included, rounds to the specified fractional scale. Midpoint rounding parameter (/mode) can be included for greater control.

- **Parameters:**



- **value:** the number to round
- **scale** (optional): the fractional digits to the right of the decimal – a number between 0 & 28
- **mode** (optional): can be *ToEven* or *AwayFromZero* – more information available at [MSDN](#)
- **Examples:**
 - *Round /value=3.45*
 - *Round /value=3.45 /scale=1*
 - *Round /value=3.45 /scale=1 /mode=AwayFromZero*
 - *concat("Round /value=3.45 /scale=", fieldWithScale)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Sin

Returns the sine of the angle provided.

- **Parameters:**
 - **value:** an angle, measured in radians or degrees - for degrees, include */input=deg*
 - **input** (optional): can be *rad* or *deg* to indicate whether */value* is radians or degrees – default is radians
- **Examples:**
 - *Sin /value=.25*
 - *Sin /value=45 /input=deg*
 - *concat("Sin /value=", fieldWithValue, " /input=deg")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Sinh

Returns the hyperbolic sine of the angle provided.

- **Parameters:**
 - **value:** a hyperbolic angle
- **Examples:**
 - *Sinh /value=.25*
 - *concat("Sinh /value=", fieldWithValue)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Sqrt

Returns the square root of the value provided.

- **Parameters:**
 - **value:** number to return square root for
- **Examples:**
 - *Sqrt /value=9*
 - *concat("Sqrt /value=", fieldWithValue)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Tan

Returns the tangent of the angle provided.



- **Parameters:**
 - **value:** an angle, measured in radians or degrees - for degrees, include /input=deg
 - **input** (optional): can be *rad* or *deg* to indicate whether /value is radians or degrees – default is radians
- **Examples:**
 - Tan /value=.25
 - Tan /value=45 /input=deg
 - concat("Tan /value=", fieldWithValue, " /input=deg")
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Tanh

Returns the hyperbolic tangent of the angle provided.

- **Parameters:**
 - **value:** a hyperbolic angle
- **Examples:**
 - Tanh /value=.25
 - concat("Tanh /value=", fieldWithValue)
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Misc or Email

JsonToXml

Converts a JSON string to XML and places the converted XML into one of the form's data sources.

- **Parameters:**
 - **xpath:** location where the converted XML should be inserted. This should be the path to an XML element.
 - **json:** the JSON string to convert
 - **dsname** (optional): data source where the converted XML should be inserted.
- **Examples:**
 - `JsonToXml /json-xpath=/*/TicketResult/@jsonContent /json-dsname=ConvertedXML /dsname=ConvertedXML /xpath=/*/TicketResult`
- **Additional Observation:**
 - The converted XML is inserted as a child of the element specified by the /xpath parameter.
 - All of the element's existing child nodes are removed before doing so.
 - This command does not currently provide a way to add XML namespaces to the converted XML, so it will not usually be possible to insert the converted XML into the main data source.
 - To avoid command parsing issues due to the content of the JSON value, you can use /json-xpath and /json-dsname to specify the location of a field containing the JSON string, instead of using /json
 - Click [here](#) for details on the conventions that this command uses when converting XML to JSON.

LoadResource



FormsViewer only command that allows loading a CSS or JS file into the page.

- **Parameters:**
 - **name:** filename of the file within the form's resource files, or the URL to the file
 - **type** (optional): either "script" or "style". Autodetermined based on the file extension if unspecified
 - **unload** (optional): Boolean, defaults to false. Unloads the file from the page. Generally only has any observable effect with CSS files
 - **allowexternal** (optional): Boolean, defaults to false. When true, allows loading a resource from a URL. If false, the command only allows loading files that are within the form.
- **Examples:**
 - *LoadResource /name=MyCss.css*
 - *LoadResource /name=http://server/site/siteassets/UpdatePage.js /allowexternal=true*
 - *LoadResource /name=MyFile /type=script*

MakeRequest

Carries out a network request and provides the result

- **Parameters:**
 - **path:** location (e.g. URL) of the request to make. Currently only http(s) URLs are supported.
 - **username** (optional): username to use for authenticating the request
 - **password** (optional): password to use for authenticating the request
 - **method** (optional): HTTP method to use (e.g. POST, GET, PUT)
 - **contenttype** (optional): MIME Content-Type of the request. Only relevant when /content is specified. Defaults to text/plain when unspecified.
 - **content** (optional): The body of the request
- **Examples:**
 - *concat("MakeRequest /path=https://", HostName, "/api/v2/tickets.json /content-dsname=ZenDesk /content-xpath=/*/TicketRequest/@jsonContent /username=", UserName, "/password=", Password, " /contenttype=application/json /resultdestds=ZenDesk /resultxpath=/*/TicketResult/@jsonContent")*
- **Additional Observation:**
 - The request's response, if any, is decoded as text and placed in the qRules Result field.
 - This command will often require the form to be full trust if you are making requests outside of the domain where the form is published.
 - To avoid command parsing issues due to the request content, you can use /content-xpath and /content-dsname to specify the location of a field containing the request content, instead of using /content

SendEmail

Send e-mails through Outlook or an email data connection.

- **Parameters:**
 - **to:** The address(es) to which the e-mail should be sent
 - **cc** (optional): CC address if not using address set in data connection.
 - **bcc** (optional): Bcc address if not using address set in data connection.
 - **subject** (optional): Subject if not using subject set in data connection.
 - **body** (optional): Body text for email.
 - **views** (optional): List of views to render and include in the body of the email, separated by ampersands (&). Only applicable when sending via Outlook.



- **intro** (optional): Introduction if not using introduction set in data connection.
- **submit** (optional): Name of email submit data connection
- **method**: Mechanism to use in sending the e-mail. Possible values are (connection, outlook); default is 'connection'
- **attachments** (optional): XPath to base64 attachments to attach to the e-mail. Only applicable when sending via Outlook.
- **Examples:**
 - *SendEmail /to=EmailAdresss /method=outlook*
 - *concat("SendEmail /method=outlook /to=", EmailAddress)*
 - *SendEmail /to=EmailAddress /method=outlook /cc=EmailAddress /bcc=EmailAddress /subject=Subject of the email /body=body text for the email /views=View1&View2*

XmlToJson

Converts XML in one of the form's data sources to a JSON string. Useful in combination with the MakeRequest command for interacting with JSON web APIs.

- **Parameters:**
 - **xpath** (optional): XPath to the XML node to convert to JSON. If unspecified, the data source's root node is used.
 - **dsname** (optional): Name of the data source containing the node to convert to JSON. If unspecified, the main data source is used.
- **Examples:**
 - *XmlToJson /dsname=ZenDesk*
 - *XmlToJson /dsname=ZenDesk /xpath=/*/TicketRequest /resultdestds=ZenDesk /resultxpath=/*/TicketRequest/@jsonContent*
- **Additional Observation:**
 - Places the converted JSON into the qRules Result field.
 - Click [here](#) for details on the conventions that this command uses when converting XML to JSON.

Node

Copy

Copies xml nodes to a new destination.

- **Parameters:**
 - **sourcepath**: The xpath of the node to copy
 - **destpath**: The XPath of the location where the node should be copied
 - **location**: The location relative to the destination node where the node should be inserted (firstChild, lastChild, before, or after)
 - **sourcedds** (optional): The name of the data source from which the node should be copied
 - **destds** (optional): The name of the data source to which the node should be copied
- **Examples:**
 - *Copy /sourcepath=/my:CopyMoveExample/my:Table1/my:Row[1] /destpath=/my:CopyMoveExample/my:Table2 /location=firstChild*
 - *Copy /sourcepath=/my:CopyMoveExample/my:Table1/my:Row /destpath=/my:CopyMoveExample/my:Table2/my:Row[2] /location=after*



CopyRichText

Copies a rich text field. You can specify a destination and a source data source as well as the XPath to a rich-text field you wish to copy.

- **Parameters:**
 - **xpathsrc:** the XPath to the source rich text field
 - **xpathdest:** the XPath to the destination rich text field
 - **dsnamesrc** (optional): Name of the source data source. Assumes you are using the Main data source if it is not specified.
 - **dsnamedest** (optional): Name of the destination data source. Assumes you are using the Main data source if it is not specified.
- **Examples:**
 - *CopyRichText /xpathsrc=/my:myFields/my:CopyRichText/my:RT1 /xpathdest=/my:myFields/my:CopyRichText/my:RT2*
 - *CopyRichText /xpathsrc=/my:myFields/my:CopyRichText/my:RT1 /dsnamedest=DBXLSubmit /xpathdest=/dfs:myFields/dfs:dataFields/tns:SubmitDocumentResponse/tns:SubmitDocumentResult/tns:Errors/tns:ErrorInfo/tns:Description*
 - *CopyRichText /xpathdest=/my:myFields/my:CopyRichText/my:RT1 /dsnamesrc=DBXLSubmit /xpathsrc=/dfs:myFields/dfs:dataFields/tns:SubmitDocumentResponse/tns:SubmitDocumentResult/tns:Errors/tns:ErrorInfo/tns:Description*
- **Additional Observations:**
 - This command takes a path to the field you want to copy, and a path to the destination field. Therefore, using a concat statement to specify the values will not yield a result.

CopyTable

Copies data from the source table to the destination table.

- **Parameters:**
 - **dsnamesrc** (optional): name of the source data source. Assumes you are using the Main data source if it is not specified.
 - **tablesrc:** XPath to the source table
 - **rowsrc:** relative XPath from the source table to the repeating source row
 - **dsnamedest** (optional): name of the destination data source. Assumes you are using the Main data source if it is not specified.
 - **tabledest:** XPath to the destination table
 - **rowdest:** relative XPath from the destination table to the repeating destination row
 - **empty** (optional): can have a value of yes or no. If set to *yes*, the destination table will be emptied before the Copy action. If unspecified, the value defaults to *no*.
 - **xpathdest** (optional): the full, unfiltered XPath to the destination row (used when the tabledest parameter is repeating and contains filters)
- **Examples:**
 - *CopyTable /tablesrc=/my:myFields/my:group1 /rowsrc=my:group2 /tabledest=/my:myFields/my:group3 /rowdest=my:group4 /empty=yes*
 - *CopyTable /tablesrc=/my:myFields/my:group1 /rowsrc=my:group2 /tabledest=/my:myFields/my:group3[my:field1 = 'Foo'] /rowdest=my:group4 /empty=yes /xpathdest=/my:myFields/my:group3*
 - *concat("CopyTable /dsnamesrc=MySharePointList /tablesrc=/dfs:myFields/dfs:dataFields /rowsrc=d:SharePointListItem_RW /tabledest=/my:myFields/my:group3 /rowdest=my:group4 /empty=", fieldWithYesOrNoForEmpty)*
- **Additional Observations:**



- Attributes and rich text can be copied
- Column names within your source and destination tables do not need to match.
- When using the CopyTable in a SharePoint 2007 library or list, it will only work if the nodes match, which also means each column has to have data in it (no blank fields). The reason your fields can't be blank is because when there is a blank field, no placeholder for that field is brought down in the XML, so it's as if that node doesn't exist. This limitation is not present in SharePoint 2010 lists data connections.
- Previous to v4.3, CopyTable would sometimes discard the nillable attribute, making a field required. This has been fixed in v4.3 so CopyTable brings in the nil attribute.

Delete

Deletes an xml node specified by an XPath.

- **Parameters:**
 - **dsname:** data source (optional): only needs to be specified if the XPath to delete is not located in the Main data source.
 - **xpath:** XPath to delete. It can be filtered in the case of a repeating group or field.
- **Examples:**
 - *Delete /xpath=/my:myFields/my:field1*

GetValue

Returns the value for a given XPath as a string – if the node is not present or blank, a default value can be returned.

- **Parameters:**
 - **dsname** (optional): name of the data source for the node, assumes Main if not specified
 - **xpath:** XPath to the field with the value to get
 - **eval** (optional): Boolean. When /eval is true, it treats the /xpath parameter as an expression.
 - **default** (optional): a default value to return if the node does not exist, or returns blank
- **Examples:**
 - *GetValue /xpath=/my:myFields/my:field /default="NewValue"*
 - *concat("GetValue /xpath=", field1)*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GetXml

Returns the inner XML or outer XML for a given XPath. This can then be passed to a web service that requires XML be passed as a string value.

- **Parameters:**
 - **xpath** (optional): XPath to execute command on
 - **inner** (optional): can be yes or no
 - **outer** (optional): can be yes or no
 - **dsname** (optional): only needs to be specified if the XPath is not located in the Main data source.
- **Examples:**
 - *GetXml /xpath=/my:myFields/my:richtextfield /outer=yes*
 - *GetXml /xpath=/my:myFields/my:richtextfield /inner=yes*
- **Additional Observations:**



- If **inner** or **outer** are not specified, **outer** is used by default.
- Returns result to the QdabraRules data source Result node.

Insert

While [ExecuteAction](#) is limited to the active view, Insert is not. It allows insertion of rows in a repeating table.

- **Parameters:**
 - **dsname** (optional): name of the data source. Assumes you are using the Main data source if it is not specified.
 - **parent**: XPath to the parent node (insertion point)
 - **child** (optional): relative XPath to the child node (repeating) – must provide either /child or /xpathsrc
 - **before** (optional): relative XPath from the parent node to the sibling node that child node should be inserted before (ignored when using /posn, if node cannot be selected, simply inserts as usual)
 - **posn** (optional): row position where to insert. Assumes “last” if it is not specified.
 - **count** (optional): number of occurrences to insert. Assumes “1” if it is not specified.
 - **xpathsrc** (optional): specifies the XPath to the xml fragment to insert
 - **firstparentonly** (optional): when the form has a repeating group (child) within a repeating group (parent), the schema may have multiple parents. Executing the Insert command with /firstparentonly set to true (or not declared) means that only the first parent will receive the action specified by the Insert command. If /firstparentonly is set to false, then EVERY parent node will receive the action specified by the Insert command. Defaults to true.
- **Examples:**
 - *Insert /parent=/my:myFields/my:Insert /child=my:InsertGroup /posn=2 /count=3*
 - *concat("Insert /parent=/my:myFields/my:Insert /child=my:InsertGroup /posn=", [InsertPosition](#), " /count=", [InsertCount](#))*
 - *Insert
/parent=/my:myFields/my:ContainerGroup/my:RepeatingGroup[3]/my:ChildContainerGroup
/xpathsrc=/my:myFields/my:ContainerGroup/my:RepeatingGroup/my:ChildContainerGroup/
my:ChildRepeatingGroup*
- **Additional Observations:**
 - Either **child** or **xpathsrc** must be specified. If both are, **xpathsrc** takes precedence.
 - If you need to specify an XPath value for count, make sure to use a concat operation, as shown in the example above.

Move

Moves xml nodes to a new destination.

- **Parameters:**
 - **sourcepath**: The xpath of the node to copy
 - **destpath**: The XPath of the location where the node should be copied
 - **location**: The location relative to the destination node where the node should be inserted (firstChild, lastChild, before, or after)
 - **sourceds** (optional): The name of the data source from which the node should be copied
 - **destds** (optional): The name of the data source to which the node should be copied
- **Examples:**
 - *Move /sourcepath=/my:CopyMoveExample/my:Table1/my:Row[1]
/destpath=/my:CopyMoveExample/my:Table2 /location=firstChild*



- *Move /sourcepath=/my:CopyMoveExample/my:Table1/my:Row
/destpath=/my:CopyMoveExample/my:Table2/my:Row[2] /location=after*

MoveTable

Moves data from the source table to the destination table. This is similar to [CopyTable](#), except that the source table is not preserved.

- **Parameters:**
 - **dsnamesrc** (optional): name of the source data source. Assumes you are using the Main data source if it is not specified.
 - **tablesrc**: XPath to the source table
 - **rowsrc**: relative XPath from the source table to the repeating source row
 - **dsnamedest** (optional): name of the destination data source. Assumes you are using the Main data source if it is not specified.
 - **tabledest**: XPath to the destination table
 - **rowdest**: relative XPath from the destination table to the repeating destination row
 - **empty** (optional): can have a value of yes or no. If set to yes, the destination table will be emptied before the Move action. If unspecified, the value defaults to no.
- **Examples:**
 - *MoveTable /tablesrc=/my:myFields/my:group1 /rowsrc=my:group2
/tabledest=/my:myFields/my:group3 /rowdest=my:group4 /empty=yes*

Replace

Replaces data in the destination group with data from the source group; able to handle a more complex node structure than [CopyTable](#) or [MoveTable](#).

- **Parameters:**
 - **dsnamesrc** (optional): name of the source data source. Assumes you are using the Main data source if it is not specified.
 - **xpathsrc**: XPath to the source replacement node
 - **dsnamedest** (optional): name of the destination data source. Assumes you are using the Main data source if it is not specified.
 - **xpathdest**: XPath to the destination node being replaced
 - **nsredef** (optional): specify a namespace prefix to have the source definition replaced with the destination definition. If the prefix and definition both change, specify <old namespace>:<new namespace>. An example of this is shown below.
- **Examples:**
 - *Replace /xpathsrc=/my:myFields/my:Replace/my:ReplaceGroup1/my:group1
/xpathdest=/my:myFields/my:Replace/my:ReplaceGroup2/my:group1*
 - *Replace /dsnamesrc=externalmyx /xpathsrc=/ns1:myFields/ns1:ReplaceGroup1
/xpathdest=/my:myFields/my:ReplaceGroup1 /nsredef=myx:my*
 - *Replace /dsnamesrc=external /xpathsrc=/ns1:myFields/ns1:ReplaceGroup1
/xpathdest=/my:myFields/my:ReplaceGroup1 /nsredef=my*
- **Additional Observations:**
 - The source and the destination nodes structure must match, including group names and column names. In the above example,
/my:myFields/my:Replace/my:ReplaceGroup1/my:group1 references
/my:myFields/my:Replace/my:ReplaceGroup2/my:group1, where /mygroup1 can have a child repeating node and some fields.



- If you are replacing a node in the main data source with a node in a secondary data source, note that namespaces and structures should match exactly. Otherwise, use the [CopyTable](#) command.

SetValue

Sets the value of the specified field. Can be used to specify a particular field (like in a repeating group) using XPath filters.

- **Parameters:**
 - **xpath:** XPath to the field to set
 - **value** (optional): new value for the field being set
 - **dsname** (optional): only needs to be declared if it is not the Main data source.
 - **blank** (optional): the field gets set to blank when /blank=true, yes or 1.
 - **nil** (optional): the field gets set to blank and has the nil attribute added (xsi:nil="true") when /nil=true, yes or 1.
- **Examples:**

In the following examples, /my:group2 is a repeating group and /my:field4 is the field being set.

 - `SetValue /xpath=xyz /nil=1`
 - `SetValue /xpath=xyz /blank=1`
 - `SetValue /xpath=xyz /blank=1 /nil=1` (this is the same as: `SetValue /xpath=xyz /nil=1`)
 - `SetValue /xpath=xyz /value=abc`
 - `SetValue /xpath=/my:myFields/my:group1/my:group2/my:field4[./my:field3='a'] /value=Test`
 - `concat("SetValue /xpath=my:myFields/my:group1/my:group2/my:field4[./my:field3='", field1, "' ] /value=", field2)`
- **Additional Observations:**
 - At least one of the following parameters should be provided: /value, /blank or /nil

SortTable

Sorts a table according to the specified column(s) and order(s).

- **Parameters:**
 - **dsname** (optional): name of the data source. Assumes you are using the Main data source if it is not specified.
 - **row:** relative path to the repeating row to sort on, so should not include leading slash
 - **compare:** relative path to the field to sort on. Must be a node within the repeating row.
 - **order** (optional): can be ASC or DESC, not case sensitive. Assumes ASC if not specified.
 - **type** (optional): can be number or text. Note that if the incorrect types are used, the rule will not execute. Assumed text if not specified.
- **Examples:**
 - `SortTable /row=my:myFields/my:MyGroup /compare=my:id /order=DESC`
 - `SortTable /row=my:myFields/my:MyGroup /compare=my:id,my:name /order=DESC,ASC /type=text,text`
- **Additional Observations:**
 - If you are sorting your data on multiple columns, you will need a matching number of columns, sort orders and data types. For instance: three columns, three sort orders and three data types.
 - If you are sorting on more than one column, you will need to specify order and type. (i.e. they stop being optional)



- The lists declared when sorting on multiple columns need to be comma-delimited.
- Note that your data must contain at least two rows in to have data to sort on.
- This command does not support repeating tables that lie outside of a container group.
 - We recommend that you move the repeating table into a container group. This is the best option as it componentizes the table and the structure has some advantages in InfoPath and follows best practices for schema design.
 - One other option would be to sort in the reverse order of what was asked for, then INSERT BEFORE the first instance of the rows being sorted, deleting each row from its original location after it is inserted at the top. When all the rows have been processed, they will appear in the correct sort order.

Transform

Performs an XSL transform on data in any data source and stores the result somewhere else in the form. Supports using **xsl:include** and **xsl:import** to import other XSLTs that are stored in the form as a resource file. The resulting XML can be stored as either a string value in a single field, or as an entire node tree beneath a certain node. This command utilizes an XSLT stored as a resource file in the XSN to perform its transform. You can use the Resource Files dialog to add and manage these files. This command is recommended for experienced users who know how to create an XSLT. For more information on this, check the following links:

<http://www.w3schools.com/xsl/>

<http://www.tizag.com/xmlTutorial/xslttutorial.php>

- **Parameters:**

- **xsltfile**: name of a resource file in the XSN that contains an XSLT as its content
- **sourceds** (optional): name of the data source for the input to the transform. Assumes Main data source if unspecified.
- **sourcepath** (optional): XPath of the node to use as the root of the input document. Defaults to root of the input data source if unspecified.
- **destds** (optional): name of the data source where the output of the transform should be placed. Assumes Main data source if unspecified.
- **destpath** (optional): XPath of the node where the output of the transform should be placed. Defaults to root of the destination data source if unspecified.
- **tostring** (optional): If true, places the result of the transform in the destination node as an escaped string. If false, places the result in the destination node as a node tree, replacing any other children that node already has. Defaults to false.
- **sourcefile** (optional): allows applying a transform to one of the template's files (a resource file or any other XML file in the template). The value of /sourcefile would be the name of the file to transform.
- **excluderoot** (optional): Boolean. When /excluderoot is true, the command will insert all of the XML *beneath* the root of the transformed XML into the destination node. When it is false, the entire transformed XML is inserted into the destination node.
- **placement** (optional): Location to place the transform result in the destination node. Possible values are: replace, prepend, append (default is replace)
- **usedocument** (optional): Boolean, false by default. Allow using the document() function in the XSLT.

- **Examples:**



<http://www.qdabra.com>

Copyright © 2006-2016 Autonomy Systems, LLC. All rights reserved.

- *Transform /xsltfile=ProcessCustomersResult.xsl /sourceds=QueryDb_Customers /sourcepath=/dfs:myFields/dfs:dataFields/tns:GetColumnsXMLStrQueryResponse/tns:GetColumnsXMLStrQueryResult/Rows /destpath=/my:myFields/my:Customers*
- *Transform /sourceds=Customers /destpath=/my:myFields/my:Customers /xsltfile=ConvertCustomers.xslt /placement=prepend*
- *Transform /destpath=/my:myFields/my:Customers /xsltfile=ConvertCustomers2.xslt /usedocument=true*
- **Additional Observations:**
 - If the destination DS is the main data source, any XML being inserted must be consistent with the main data source schema.
 - This command cannot replace the root node of the main data source, but can replace everything beneath it. The output of the transform does not have to contain a single root node.
 - The XSLT must use the omit-xml-declaration directive if using /tostring=true

SharePoint

AttachToSharePointList

The command AttachToSharePointList saves attachments to list items so you can replace with a link and a filename attribute instead. The command includes logic to create a unique filename for each attachment in the list; alternatively, it allows you to provide your own name for the attachment. The command also works for repeating attachments. If you have the qRulesLink and qRulesFilename attributes, this command will also remove the attachment from your form and populate with link and filename.

- **Parameters:**
 - **dsname:** data source For AddAttachment Method (SharePoint Lists Web Service)
 - **url** (optional): SharePoint Lists Web Service URL
 - **guid** (optional): GUID for SharePoint List. As of qRules v5.1, you can also provide the list name instead of the guid via this parameter.
 - **id:** ID of list item to attach to
 - **xpath:** XPath to attachment field
 - **namexpath** (optional): xpath to the file name
 - **name**(optional): name for file (extension will be added by qRules)
 - **uniqueName** (optional): prepend date and time information to the file name to provide a unique name; can be set to 'yes' or 'no', default is no
 - **clear** (optional): yes | no - allows the user to control whether the attachment is cleared from the main data source (XML) once the upload is finished.
- **Examples:**
 - *AttachToSharePointList /dsname=AddAttachment /id=1 /xpath=/my:myFields/my:AttFolder/my:field1 /uniqueName=yes*
 - *concat("AttachToSharePointList /dsname=AddAttachment /id=", fieldWithId, " /xpath=/my:myFields/my:AttFolder/my:field1")*
 - *AttachToSharePointList /dsname=AddAttachment /id=1 /xpath=/my:myFields/my:AttFolder/my:field1*
 - *AttachToSharePointList /dsname=AddAttachment /id=1 /xpath=/my:myFields/my:AttFolder/my:field1 /name=new /uniqueName=yes*



- *AttachToSharePointList /dsname=AddAttachment /url=http://server/_vti_bin/lists.asmx?WSDL /guid= {CDAA93D1-C8D5-433F-A9EE-6FE7456CD76C} /id=1 /xpath=/my:myFields/my:AttFolder/my:field1 /name=new /uniqueName=yes*
- *AttachToSharePointList /dsname=AddAttachment /guid= {CDAA93D1-C8D5-433F-A9EE-6FE7456CD76C} /id=1 /xpath=/my:myFields/my:AttFolder/my:field1 /name=new /uniqueName=yes*
- **Additional Observations:**
 - When setting up the AddAttachments data connection, one can provide a listName in the data connection wizard. If this is entered in the data connection, then the /guid parameter is not necessary.
 - If the data connection was created using a listName parameter, AND a /guid parameter is provided, the /guid parameter will be used.
 - On SharePoint 2013 (as of June 2013):
 - Sandboxed forms must set Use Web Services to TRUE before executing this command.
 - Admin deployed forms must be full trust for this command to work.

CreateSharePointFolder

Allows users to create a folder in a SharePoint library or list.

- **Parameters:**
 - **name:** path of the new folder to create, relative to the list or library
 - **webservice:** The name of a data connection set up to use Lists.asmx's UpdateListItems method. (Required when creating folders from Filler mode.)
 - **type** (optional): Either Folder or DocumentSet. Default is Folder
 - **contenttype** (optional): A custom content type name or ID to use
 - **contenttypeisid** (optional): (Boolean) true to indicate that the /contenttype value is a content type ID (false by default)
 - **siteurl** (optional): Url of the SharePoint site where the folder or document set should be created. Required if not specified in a definition file.
 - **library** (optional): Name of the library where the folder or document set should be created. Required if not specified in a definition file.
 - **definitionsrc** (optional): Name of the data source where a custom definition is.
 - **definitionname** (optional): Name of the definition within the definition file.
- **Examples:**
 - *CreateSharePointFolder /name=Test /webservice=Submit /siteurl=http://shp2010test/ /library=FolderA*
 - *CreateSharePointFolder /definitionsrc=DefinitionsFolder /definitionname=Definition1 /name=Test1 /webservice=Submit*
- **Additional Observations:**
 - There is an additional option to create a DocumentSet (/type=DocumentSet). For assistance, please contact Qdabra support.
 - If the folder already exists, no error message is returned.
 - A definition file is an XML file, included as a secondary data source, and with a structure as shown below:


```
<Definitions schemaVersion="1.0">
  <Definition name="Definition1" type="folder">
    <Properties siteUrl="http://shp2010test/Test/" library="LibraryA" />
  </Definition>
  <Definition name="Definition2" type="folder">
```



```
<Properties siteUrl="http://shp2010test/Test/" library="LibraryB" />
</Definition>
</Definitions>
```

DeleteFromSharePoint

Allows users to delete a document in SharePoint.

- **Parameters:**
 - **url:** document to be deleted.
- **Examples:**
 - *DeleteFromSharePoint*
/url=http://<YourSharePointSite>/QdFile/2011031502554597899_Picture.jpg
 - *concat("DeleteFromSharePoint /url=", [qRulesLink](#))*
- **Additional Observations:**
 - This command is helpful for deleting files and images added to a library with [SaveToSharePoint](#).
 - This command, like [SaveToSharePoint](#), needs the form to be full trust to work in Preview mode or if deleting from a site other than that the form is published to.
 - For SharePoint 2013 browser forms, this commands require the form be full trust and admin deployed.

FilterOwssvr

Adds a filter to a data connection to owssvr.dll and executes the connection. Additional documentation for this command is available [here](#).

- **Parameters:**
 - **dsname:** data connection to the DLL file using the decoded ListID, for example, http://server/_vti_bin/owssvr.dll?Cmd=Display&List={0736ECB6-9AB0-4A95-98E7-9FC845CE2D58}&XMLDATA=TRUE
 - **filter:** list column to filter on. Note that the values in this parameter are case-sensitive.
 - **value:** value to match
 - **delimiter** (optional): single character to use as a delimiter for filter and value list in case values have commas.
- **Examples:**
 - One filter: *FilterOwssvr /dsname=FilterOwssvr /filter=Deduction /value=Charity*
 - Multiple filters: *FilterOwssvr /dsname=FilterOwssvr /filter=Deduction,Price /value=Charity,1000*
 - Null/blank value filters: *FilterOwssvr /dsname=FilterOwssvr /filter=Deduction /value="*
 - With delimiter: *FilterOwssvr /dsname=owssvr /filter=Title|Department /value=Assistant|HR /delimiter=|*
- **Additional Observations:**
 - The list must have at least two entries for InfoPath to interpret the schema correctly when creating the data connection.
 - SharePoint revises column names automatically, which may cause troubles using FilterOwssvr, so you want to check the field name in SharePoint. To do that, copy the qRules result string for FilterOwssvr (i.e. the URL that is generated, such as http://server/_vti_bin/owssvr.dll?Cmd=Display&List={1CC3F2DD-4251-4A98-9BC9-E584496C0727}&XMLDATA=TRUE), then open that URL in a web browser to see the result.



- This command is most useful on MOSS 2007. It is not supported on Office 365. Though the command is supported for SharePoint 2010, this functionality is available out of the box if you are working with InfoPath 2010 and a SharePoint 2010 list; these data connections now have query fields.
- In InfoPath 2010, this command is no longer needed, as you can create a REST data connection to the owssvr.dll and then use rules to add the filters.

GetListGuid

Uses a data connection to the Lists.asmx web service GetListCollection method to return the GUID for the list named in \listname.

- **Parameters**
 - **dsname**: The data connection to the GetListCollection method of the Lists web service.
 - **listname**: The name of the list to return the GUID for
- **Example**
 - *GetListGuid /listname=PedidosRateio /dsname=GetListCollection*
- **Additional Observations:**
 - Requires Use Web Services = TRUE in SharePoint 2013.

GetTermPath

Queries SharePoint's TaxonomyClientService to obtain the path for a given Term. Additional information on the metadata commands can be found [here](#).

- **Parameters:**
 - **termid**: the term id whose path you wish to obtain
 - **pathsep** (optional): used to establish a delimiter other than semicolon.
- **Examples:**
 - The command is probably going to be called after obtaining a term via the GetTermSets command. Therefore, use concat to use that term:
concat("GetTermPath /termId=", term)
- **Additional Observations:**
 - Be sure to check the "Inject Term Set support" when injecting qRules if you wish to use this command.
 - qRules' Managed Metadata commands are not available on Office 365.

GetTermsByLabel

Queries SharePoint's TaxonomyClientService to obtain a list of TermSets that match the label (search term) provided by the user. Additional information on the metadata commands can be found [here](#). This command takes no parameters.

- **Parameters:**
 - none
- **Examples:**
 - GetTermsByLabel
- **Additional Observations:**
 - Be sure to check the "Inject Term Set support" when injecting qRules if you wish to use this command.
 - qRules' Managed Metadata commands are not available on Office 365.



GetTermSetIds

Queries SharePoint to obtain a list of available TermSetIds. Additional information on the metadata commands can be found [here](#).

- **Parameters:**
 - **list** (optional): name of list to containing managed metadata columns. Defaults to Shared Documents if not specified.
 - **dsnamesrc** (optional): name of GetList data connection. Defaults to GetList if not specified.
 - **dsnamedest** (optional): name of GetTermSet data connection. Defaults to GetTermSet if not specified.
 - **timeout** (optional): connection timeout
- **Examples:**
 - *GetTermSetIds*
 - *GetTermSetIds /list=GetList*
- **Additional Observations:**
 - Be sure to check the "Inject Term Set support" when injecting qRules if you wish to use this command.
 - qRules' Managed Metadata commands are not available on Office 365.

GetTermSets

Queries SharePoint's TaxonomyClientService to obtain a list of available TermSets. Additional information on the metadata commands can be found [here](#). This command takes no parameters.

- **Parameters:**
 - none
- **Examples:**
 - *GetTermSets*
- **Additional Observations:**
 - Be sure to check the "Inject Term Set support" when injecting qRules if you wish to use this command.
 - qRules' Managed Metadata commands are not available on Office 365.

GetUserName

Retrieves the current user's alias.

- **Parameters:**
 - none
- **Examples:**
 - *GetUserName*
- **Additional Observations:**
 - While in Preview, MOSS 2007 and SharePoint 2010, the username is an easy-to-read string, in SharePoint 2013 and Office 365 you'll see usernames like i:0#.w|<domain>\<username> and i:0#.f|membership|<emailAddress>, respectively.

GetUserProfileByName

Retrieves the user information from SharePoint's GetUserProfileByName web service.

- **Parameters:**



- **dsname** (optional): name of the data connection to SharePoint's GetUserProfileByName method (SharePoint UserProfileService Web Service). Defaults to GetUserProfileByName if not specified.
- **user** (optional): the user profile will be retrieved for the user specified. Assumes the current logged in user if not specified.
- **Examples:**
 - *GetUserProfileByName /user=mikesmith*
 - *concat("GetUserProfileByName /user=", fieldWithUserName)*
- **Additional Observations:**
 - The form must have a data connection to the GetUserProfileByName web method in */_vti_bin/UserProfileService.aspx* in order for this command to work. This data connection must not execute on load.
 - The information returned in Office 365 will be limited.
 - For SharePoint 2013 browser forms, this commands require the form be full trust and admin deployed.

RefreshSharePointListItems

This command is used in tandem with SubmitToSharePointList to update list items with most recent edits to the list. Supports changes, including deletes. Can be used in either report or update mode.

This command, like SubmitToSharePoint, underwent major revisions with qRules v4.2. The command syntax as documented below works with qRules v4.2 and later, and requires the use of the updated mapping form that installs with qRules v4.2. Previous versions of the command will continue to work without any changes. More information can be found [here](#). Additional information on the structure of the mapping file can be found [here](#).

- **Parameters:**
 - **dsname**: data source name for list data
 - **id** (optional): XPath to ID field(s) for SharePoint List ID (relative path for repeating). Must be included if ID was not specified in the mapping file.
 - **mapping** (optional): mapping data source name. Assumes *mapping* if not specified. Mapping files can be created with the InfoPath to SharePoint list tool that comes with qRules. Information on installing that full trust form can be found [here](#).
 - **mappingName** (optional): specify the mapping to refresh, refreshes all mappings in mapping file if not specified.
 - **mode** (optional): can be Report or Update. Default is Report
 - **xpathsrc** (optional): path to the repeating node in your SharePoint list where you have your list items you want to use for refreshing your form. When using this parameter, dsname will be a data connection to the SharePoint List, including the *Modified* column. The form will need to query for data prior to executing the Refresh command if using this option, as the command will not execute the query. If using this parameter, /mappingName is required.
- **Examples:**
 - *RefreshSharePointListItems /dsname=owssvr /mapping=testmapping /mode=Update*
 - *RefreshSharePointListItems /dsname=owssvr /mappingName=Tasks*
 - *concat("RefreshSharePointListItems /dsname=refresh /mode=", fieldWithMode)*
 - *RefreshSharePointListItems /dsname=RefreshList /mode=Update /xpathsrc=/dfs:myFields/dfs:dataFields/d:SharePointListItem_RW /mappingName=Test*



- **Additional Observations:**

- To use this command, form data must be storing the list item ID from the SubmitToSharePointList command. You need to add an attribute to your node where you are storing your SharePoint List ID. That attribute must be named qRulesLastModified and it must be a text attribute.
- The general concept is that when your list items are submitted to SharePoint, the SubmitToSharePointList command will populate your ID field with the list item ID (connecting the item to the list) and will also populate the qRulesLastModified attribute with the last modified date and time for the item.
- The next time you open the XML, you would be able to leverage the RefreshSharePointListItems command. It uses an owssvr data connection from your form to query for any list items that have a value in the field specified as the ID in the mapping file. The command compared the last modified date for the items in the XML to the last modified for the SharePoint list. Anything where the last modified in the XML is less than the last modified in the SharePoint List is considered out of date.
- There are two ways this command can be run. The default is Report. The results node will contain information about items that are considered changed or that have been deleted from the SharePoint List. If you use /mode=Update, the XML will be updated from the ShP list – if items in the XML no longer exist in the list, they are deleted from the XML. If the items in the XML are older than the items in the list, they are updated using the mapping file provided.
- If you run the refresh command right after your form opens, and then run a SubmitToSharePointList command at a later point, you can also propagate deletes – that is, if you delete an item from your XML that had been submitted to a SharePoint List, that item will then be deleted from the list.
- You can also run the refresh command at any point in report mode to compare your list items in your XML to the list items in SharePoint.
- You can map a filtered group in the mapping tool for the repeating group – i.e. *my:myFields/my:group1/my:group2[my:approved='true']*. Only rows that match the filter will be submitted or refreshed.
- If using this command alongside Use Web Services = 'true', do not set useName="true" in the list mapping. The list guid in the mapping file must match the list guid in the owssvr.dll data connection.
- In SharePoint 2013, sandboxed forms require Use Web Services being set to true, while admin deployed forms require being full trust.

SaveToSharePoint

Saves a file, a set of files, an image or a set of images to a SharePoint Document Library. Additional information can be found [here](#).

- **Parameters:**

- **url:** URL to SharePoint document library
- **dsname** (optional): name of data source - only needs to be specified if the XPath to the attachment is not located in the Main data source.
- **xpath:** XPath to file/image attachment
- **name** (optional): name for file (extension will be added by qRules)
- **overwrite** (optional): overwrite existing file if name is specified, can be yes or no. Default is no. Not available if /dsname is specified.
- **async** (optional): can be yes or no – default is no. Full trust only.



- **dsnamews** (optional): name of data connection to CopyIntoItems method (SharePoint Copy Web Service)
- **dsurl** (optional): if the dsnamews parameter is used, the CopyIntoItems method requires that the destination be at the same site as the data connection uses. If you would like to dynamically change the web service url for the Copy web service connection, the dsurl parameter can be used.
- **namexpath** (optional): relative xpath to the attribute which stores the desired filename (in the case of a repeating structure).
- **clear** (optional): yes | no - allows the user to control whether the attachment is cleared from the main data source (XML) once the upload is finished.
- **uniquename** (optional): Boolean parameter that defaults to true if unspecified. When it is true, qRules will append the current date and time to the file name to ensure that it is unique. When false, qRules will use the actual file name, which can cause collisions if a file with that name already exists in the target location.
- **timeout** (optional): timeout length, in seconds. Only used with Filler version of command.
- **Examples:**
 - *SaveToSharePoint /url=http://<servername>/MyDocLib /xpath=/my:myFields/my:File*
 - *SaveToSharePoint /url=http://<servername>/MyDocLib /xpath=/my:myFields/my:Images/my:Image*
 - *SaveToSharePoint /url=http://server/site/docLibrary /xpath=/my:myFields/my:Images/my:Image /dsnamews=Copy /dsurl=http://anotherServer/site/*
 - *SaveToSharePoint /url=http://<servername>/MyDocLib /xpath=/my:myFields/my:Images/my:Image /timeout=120*
- **Additional Observations:**
 - Source field must have **qRulesLink** and **qRulesFilename** attributes (please refer to the **qRules User Guide** for details).
 - If uploading more than one file/image, the source field must be repeating (not a repeating field in a repeating group (please refer to the **qRules User Guide** for details).
 - This command can work without SharePoint, by saving files to an ordinary IIS website. [Click here for instructions.](#)
 - qRules will perform no action if the user calls the rule with no image/file attached. When designing your InfoPath solution, you can design your form so the rule only executes when the image/file attachment is not empty (blank).
 - The InfoPath form and the SharePoint Document Library must live on the same machine.
 - The /async parameter allows the user to keep working in the form while the file is saved, but is only available in full trust, filler forms.
 - If using the /async option, be aware that at the completion of the command, the Result and Error nodes will be set with the outcome of the /async command. If other qRules commands have been run in the duration of the /async command, the Result or Error node may no longer be accurate for that command.
 - For SharePoint 2013 browser forms, this commands require the form be full trust and admin deployed.

SubmitAsUser

Submits an XML document to a SharePoint form library.

- **Parameters:**
 - **dsname:** SharePoint Submit Data Source Name. This is just a regular submit type data connection to a form library.



- **username:** User to impersonate for submit
- **overwrite** (optional): Boolean. Overwrite existing file if it exists. Default is false.
- **Examples:**
 - *SubmitAsUser /dsname=Submit /username=domain\johndoe*
- **Additional Observations:**
 - Only available in SharePoint 2010 and 2013 admin deployed forms. If you do not inject with the Admin Deploy option, the command will return an error.
 - When used in Filler (Rich Client), SubmitAsUser cannot overwrite and cannot impersonate user.
 - As of qRules v4.3, SubmitAsUser is not compatible with MOSS 2007 browser forms.

SubmitToSharePoint

Submits an XML document to a SharePoint form library.

- **Parameters:**
 - **dssubmit:** Sharepoint Library Submit data connection
 - **dsname** (optional): Data source that will be submitted. Defaults to the form's main data source.
 - **url** (optional): folder location to save. If not provided, the command will use the submit location defined in the Submit data connection (dssubmit).
- **Examples:**
 - *SubmitToSharePoint /dssubmit=ShPSubmit*
 - *concat("SubmitToSharePoint /dssubmit=ShPSubmit /url=", fieldWithUrl)*

SubmitToSharePointList

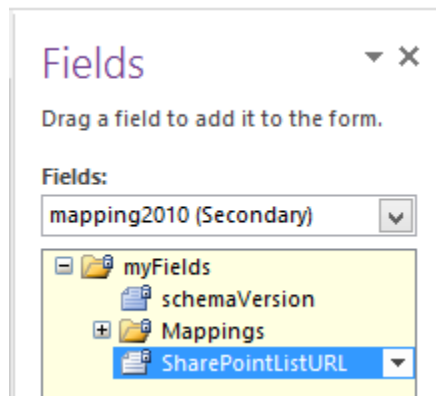
Submits data to the SharePoint list it has been mapped to. If provided with the path to a field for the list item ID, populates that, allowing for updating of items in the list from the form. Requires use of **InfoPath to SharePoint List Tool** for mapping.

This command, like [RefreshSharePointListItems](#), underwent major revisions with qRules v4.2. The command syntax as documented below works with qRules v4.2 and later, and requires the use of the updated mapping form that installs with qRules v4.2. Previous versions of the command will continue to work without any changes. More information can be found [here](#). Additional information on the structure of the mapping file can be found [here](#). A walkthrough blog [post](#) is also available.

- **Parameters:**
 - **submit:** submit data connection to UpdateListItems Method (SharePoint Lists Web Service)
 - **mapping** (optional): mapping data source name. Assumes **mapping** if not specified. Mapping files can be created with the InfoPath to SharePoint list tool that comes with qRules. Information on installing that full trust form can be found [here](#).
 - **mappingName** (optional): specify the mapping to refresh, refreshes all mappings in mapping file if not specified.
 - **id** (optional): XPath to ID field(s) for SharePoint List ID (relative path for repeating). If submitting a mapping with multiple lists, this must be indicated in the mapping file instead.
 - **dsname** (optional): data source for AddAttachment Method (SharePoint Lists Web Service) – include if submitting attachments with your list items.



- **xpath** (optional): XPath to attachment field. If submitting a mapping with multiple lists, this must be indicated in the mapping file instead.
- **name** (optional): name for attachment file (extension will be added by qRules).
- **uniqueName** (optional): prepend date and time information to the file name to provide a unique name (yes | no)
- **namexpath** (optional): When mapping more than one attachment, the namexpath allows specifying the filename that will be used for each attachment. The namexpath parameter will take a relative xpath from the attachment field.
- **clear** (optional): yes | no - allows the user to control whether the attachment is cleared from the main data source (XML) once the upload is finished.
- **rootFolder** (optional): For submitting list items to a folder. The value can be copied from the URL to the folder you wish to submit to – open the folder in the browser, copy out the URL and look for the RootFolder query parameter.
- **Examples:**
 - *SubmitToSharePointList /submit=ShPSubmit*
 - *SubmitToSharePointList /submit=ShPSubmit /mapping=testmapping /rootFolder=/sites/ProductTesting/qRulesSanityCheck/Lists/MappingList/test*
 - *SubmitToSharePointList /submit=ShPSubmit /dsname=AddAttachment*
 - *concat("SubmitToSharePointList /submit=ShPSubmit /mappingName=", fieldWithNameOfMappingToSubmit)*
- **Additional Observations:**
 - Use with [RefreshSharePointListItems](#) for full list synching.
 - When migrating a form to a new location, or using the same form in various locations, the form rules will need to update the URL in the mapping file. The line in the mapping file looks like this: **<SharePointListURL>http://shp2010-test2/qRulesv52/</SharePointListURL>** This node can be updated via form rules by setting the value of this node:



- If an xpath is not found, the column in SharePoint will be blank.
- You can map a filtered group in the mapping tool for the repeating group– i.e. *my:myFields/my:group1/my:group2[my:approved='true']*. Only rows that match the filter will be submitted or refreshed.
- When using this command in SharePoint 2010, using an InfoPath 2010 browser form that has been deployed via Sandbox, the SharePoint list will show the System Account as the creator of the item. If this is not acceptable, please Admin-deploy your form instead of using the sandbox.
- If submitting from one repeating group to two lists, only one of the mappings can make use of the /id parameter.



- If using an InfoPath 2007 browser form, admin deployed, you will need to set useWebServices to TRUE.
- In previous versions of qRules, a SharePoint list mapping would use the list GUID. Starting in v4.3, the mapping can use the list name instead, which makes moving the form easier.
- In SharePoint 2013, sandboxed forms require Use Web Services being set to true, while admin deployed forms require being full trust.
- In the InfoPath to SharePoint List Tool (aka the Mapping Tool), you'll find a checkbox to identify rich text columns. If mapping form data into a rich text column in SharePoint, this checkbox must be used. Some formatting issues will arise if there are mismatches between plain and rich text fields and columns.

String

CompareStrings

Compares two strings using the comparison type provided. Returns an integer indicating the lexical relationship (relative position in the sort order) - zero implies equivalency.

- **Parameters:**
 - **dsname1** (optional): data source name - only needs to be specified if the field selected with the /xpath1 parameter is not located in the Main data source.
 - **xpath1**: XPath to the first field for the comparison
 - **dsname2** (optional): data source name - only needs to be specified if the field selected with the /xpath2 parameter is not located in the Main data source.
 - **xpath2**: XPath to the second field for the comparison
 - **compType** (optional): string comparison type, defaults to CurrentCultureIgnoreCase. More information is available at MSDN [here](#).
- **Examples:**
 - *CompareStrings /xpath1=/my:myFields/my:firstString /xpath2=/my:myFields/my:secondString*
 - *CompareStrings /dsname1=Secondary /xpath1=/Test/TestNode /xpath2=/my:myFields/my:secondString*
 - *CompareStrings /xpath1=/my:myFields/my:firstString /xpath2=/my:myFields/my:secondString /compType=CurrentCulture*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

DelimitedList

Concatenates the values of a repeating field (or field in a repeating group), separating each value with the specified delimiter.

- **Parameters:**
 - **dsname** (optional): data source name - only needs to be specified if the XPath is not located in the Main data source.
 - **xpath**: XPath to the fields to select
 - **separator** (optional): separator desired, such as comma, colon, or pipe. Defaults to semicolon.
 - **space** (optional): can be set to "before", "after" or "both" to insert additional spaces. Can be set to "delimiter" to use a space for the delimiter instead of a character.
- **Examples:**



- *DelimitedList /xpath=/my:myFields2/my:DelimitedTextFld /separator=| /space=both*
- *DelimitedList /xpath=/my:myFields/my:group1/my:group2/my:field2 /space=delimiter*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Encode

Returns an encoded or decoded string, or in the case of sharePointFilename, a string that is safe for use in a SharePoint file name, replacing unsafe characters with underscores.

- **Parameters:**
 - **text:** text to encode
 - **decode** (optional): decodes the text, can be set to true or false, defaults to false.
 - **type** (optional): encoding type, can be set to url, html, uri, uriComponent or sharePointFilename - defaults to url
- **Examples:**
 - *Encode /text=http://someplace/some where /type=url*
 - *concat("Encode /text=", field1, " /decode=true /type=uri")*
 - *concat("Encode /text=", field1, " /type=sharePointFilename")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - Cannot decode for type sharePointFilename
 - In InfoPath 2013, you may get a security exception when the Encode command is used with /type=html. The current solution is to make the form Full Trust.
 - The uriComponent option was added in v6.1, to allow encoding URI strings that are not actual URIs. It uses Uri.EscapeDataString: <http://msdn.microsoft.com/en-us/library/system.uri.escapedatastring.aspx>

EncodeBase64

Encodes the data in the specified xpath.

- **Parameters:**
 - **dsname** (optional): Name of the data source with data to convert into Base64 (default is main data source)
 - **path** (optional): Path of the data to convert into Base64 (default is the root node)
 - **asxml** (optional): (Boolean) true to convert the raw XML value to Base64, false to convert the text value (true by default)
 - **filename** (optional): Filename to use for the converted data (default is file.xml or file.txt)
 - **excludesourceroot** (optional): (Boolean) true to exclude the outer node of the source data from the converted value (false by default)
- **Examples:**
 - *EncodeBase64 /dsname=secondaryxml /excludesourceroot=true /path=/secondaryxml/helloworld*
 - *EncodeBase64 /dsname=secondaryxml /excludesourceroot=true /path=/secondaryxml/helloworld /asxml=false /filename=file.xml*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

Encrypt / Decrypt



This command transforms texts and attachments into a set of random characters and provides security by using a password to protect data. Uses the Rijndael symmetric encryption algorithm and a 256 bit key length. **Be sure to save your password(s)—qRules does not record them.**

- **Parameters:**

- **xpath:** XPath to the field to encrypt/decrypt
- **dsnamepass** (optional): data source name for the password field if it is in a secondary data source
- **pass:** XPath to the password field
- **clear** (optional): can have a value of *yes* or *no*. If set to *yes*, the password will be cleared after encryption/decryption. If unspecified, the value defaults to *yes*. (As of qRules v5.1, this parameter accepts true/false and 1/0).
- **pad** (optional): can have a value of *yes* or *no*. Only valid for Encrypt. Defaults to *no*.

- **Examples:**

- *Encrypt /xpath=/my:myFields/my:textEncrypt /pass=/my:myFields/my:password /pad=yes*
- *Encrypt /xpath=/my:myFields/my:imageEncrypt /pass=/my:myFields/my:password /pad=yes /clear=no*
- *Decrypt /xpath=/my:myFields/my:fileEncrypt /dsnamepass=myEncryption /pass=/myEncryption/Password*
- *Decrypt /xpath=/my:myFields/my:group1/my:group2/my:field1[../my:field2='a'] /pass=/my:myFields/my:password*

- **Additional Observations:**

- Encrypting/Decrypting file or image attachments will make the control appear empty in the form because it can't read the filename and extension from the string any longer. However, if /pad is set to "yes", the encrypted string will always start with "qdEncrypted". Conditional formatting can be based off that to hide a control if the node starts with qdEncrypted, and display an expression box instead.
- Fields that use whole number or decimal as data types can be encrypted/decrypted; but since we are returning a string, the field will return a data validation error.
- You cannot encrypt/decrypt a secondary data source field because every time we pull data into a form from a secondary data source, it would be the actual data. Encrypting it would only be until the data was refreshed, since it isn't stored with the form.
- In InfoPath 2010 and later, if the encrypted field is base64 (a picture or file control) and /pad=yes, the control may have a validation error upon encryption. Thus, if you wish to encrypt a base64 field and make use of the padding of "qdEncrypted" for conditional formatting, you will also want to use the [ClearErrors](#) command to clear any possible validation errors on the file or picture control
- Qdabra also has a template part available you can use to add this functionality to your form. You can download the template part and a "dots" font [here](#). A blog post outlining the use of the template part can be found [here](#).

GenerateGuid

Returns a new GUID in the format specified.

- **Parameters:**

- **format** (optional): format for the generated GUID. Default is D. Format can be:
 - D (default) = 32 digits separated by hyphens: xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxx.
 - N = 32 digits: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 - B = 32 digits separated by hyphens, enclosed in brackets: {xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx}



- P - 32 digits separated by hyphens, enclosed in parentheses: (xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx)
- **Examples:**
 - *GenerateGuid /format=N*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

GetIndex

Returns the first or last index of a string in the source node – returns -1 if the string is not found, and 0 if the source node is blank. Useful in tandem with [RemoveFromString](#) – could be used to get the value for the /start parameter for that command.

- **Parameters:**
 - **dsnamesrc** (optional): only needs to be specified if the field selected with the /xpathsrc parameter is not located in the Main data source.
 - **xpathsrc**: XPath for the source node
 - **string**: value to find
 - **pos** (optional): can have a value of *First* or *Last*, defaults to *First*. Determines whether to return the first or last index of the string in the source node.
- **Examples:**
 - *GetIndex /xpathsrc=/my:myFields/my:field1 /string=abcde*
 - *GetIndex /dsnamesrc=Secondary /xpathsrc=/Test/TestNode /string=abcde*
 - *concat("GetIndex /xpathsrc=/my:myFields/my:field1 /string=", fieldWithString, " /pos=Last")*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - Case sensitive – if the value of the source node contains the word "test" and the /string1 parameter is "TEST", no characters will be replaced.

PadString

Returns a padded string – pads the string in the source node using the character specified to the total width specified. Padding is added to the specified side. While you can exclude the /pad parameter in order to pad with whitespace, InfoPath views will generally strip the whitespace from the XML – save the XML and view in a text editor to see the added space.

- **Parameters:**
 - **dsnamesrc** (optional): data source name - only needs to be specified if the field selected with the /xpathsrc parameter is not located in the Main data source.
 - **xpathsrc**: XPath for the source node
 - **pad** (optional): character to pad with – defaults to whitespace
 - **width**: total width for padded result – that is, if the original string is 2 chars, and the /width parameter is equal to 4, the returned result will be 4 chars.
 - **side** (optional): can have a value of *Right* or *Left*, defaults to *Left*. Determines which side of the string to pad.
- **Examples:**
 - *PadString /xpathsrc=/my:myFields/my:field /pad=0 /width=10*
 - *PadString /xpathsrc=/my:myFields/my:field /pad=. /width=20 /side=Right*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.



RemoveFromString

Returns a string with the characters removed beginning at the index provided in the `/start` parameter, through the length or through `/count`, if provided. The [GetIndex](#) command could be used to get the value needed for the `/start` parameter. If a `/count` value is provided that is greater than the length of the string less the index provided in the `/start` parameter, the length from the `/start` parameter will be removed.

- **Parameters:**
 - **dsnamesrc** (optional): data source name - only needs to be specified if the field selected with the `/xpathsrc` parameter is not located in the Main data source.
 - **xpathsrc**: XPath for the source node
 - **start**: zero based position to begin deleting characters from the value in `/xpathsrc`
 - **count** (optional): number of characters to remove. If not provided, removes to end of string from `/start` parameter
- **Examples:**
 - `RemoveFromString /xpathsrc=/my:myFields/my:field /start=4`
 - `RemoveFromString /xpathsrc=/my:myFields/my:field /start=4 /count=6`
 - `concat("RemoveFromString /xpathsrc=/my:myFields/my:field /count=", fieldWithCount)`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

ReplaceString

Returns a string with the instances of `/string1` in the source node replaced with `/string2`. Can also be used with a regular expression and the `/pattern` parameter.

- **Parameters:**
 - **dsnamesrc** (optional): data source name - only needs to be specified if the field selected with the `/xpathsrc` parameter is not located in the Main data source.
 - **xpathsrc**: XPath for the source node
 - **string1** (optional): value to be replaced – either `/string1` or `/pattern` must be provided
 - **string2** (optional): value to replace with. If not supplied, a blank character is used for the replacement.
 - **pattern** (optional): regular expression pattern for use instead of `/string1`
 - **reldes** (optional): XPath that is relative to the value of the `/xpathsrc` parameter, and when `/reldes` is specified, qRules will iterate through all nodes that match `/xpathsrc`, perform the string replacement, and place the result in the field indicated by `/reldes`.
 - **append** (optional): Boolean. When it is true, the result of the string replacement is appended to the field indicated by `/reldes` instead of overwriting it.
- **Examples:**
 - `ReplaceString /xpathsrc=/my:myFields/my:field /string1=TESTING /string2=test`
 - `ReplaceString /dsnamesrc=Secondary /xpathsrc=/Test/TestNode /string1=TESTING /string2=test`
 - `ReplaceString /xpathsrc=/my:myFields/my:field /pattern=(\d+);#`
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - Case sensitive – if the value of the source node contains the word “test” and the `/string1` parameter is “TEST”, no characters will be replaced.



SetCase

Returns a string with the value of the source node in the case requested. Options are Lower, Upper and Title.

Title case sets all words to lower case, then sets the first letter to each to upper case – unless the word is in all upper case already.

So, this: *NASA sent a man to the moon*

Would become this: *NASA Sent A Man To The Moon*

And this: *Old McDonald had a farm*

Would become this: *Old Mcdonald Had A Farm*

- **Parameters:**
 - **dsnamesrc** (optional): data source name - only needs to be specified if the field selected with the /xpathsrc parameter is not located in the Main data source.
 - **xpathsrc**: XPath for the source node
 - **case** (optional): can be Upper, Lower, or Title. Default is Upper.
- **Examples:**
 - *SetCase /xpathsrc=/my:myFields/my:field*
 - *SetCase /xpathsrc=/my:myFields/my:field /case=Title*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.

TrimString

Returns a string with the value of the source node trimmed on the side specified. Trims whitespace by default but can also trim characters if the /chars parameter is provided.

- **Parameters:**
 - **dsnamesrc** (optional): data source name - only needs to be specified if the field selected with the /xpathsrc parameter is not located in the Main data source.
 - **xpathsrc**: XPath for the source node
 - **chars** (optional): characters to trim from string – default is whitespace
 - **side** (optional): can be Right, Left, or Both – default is Both. Determines which side of the source node string to trim.
- **Examples:**
 - *TrimString /xpathsrc=/my:myFields/my:field /chars=123*
 - *TrimString /xpathsrc=/my:myFields/my:field /chars=123 /side=Left*
- **Additional Observations:**
 - Returns result to the QdabraRules data source Result node.
 - Case sensitive if specifying /chars.



COMMAND COMPATIBILITY MATRIX

Please refer to this matrix for a quick guide on the command's compatibility. Any command **not** specified in the table below can be published via the sandbox in SharePoint 2010 and 2013. Please note that the number of FormsViewer commands may have increased after the publication of this guide – refer to the [Release History](#) page for more information, or contact support@qdabra.com for assistance.

Commands	Filler Only	Full trust	Sandbox possible	Admin Deploy	Browser compatible	DBXL required	FormsViewer compatible
AssignDocument			•		•	•	•
ChangeConnectionUrl			•		•		•
ChangeSubmitUrl			•		•		•
CopyRichText			•		•		•
CopyTable			•		•		•
CreateDirectory	•	•					
DateAdd			•		•		•
DateDiff			•		•		•
Decrypt			•		•		•
Delete			•		•		•
DeleteFile	•	•					
DeleteFromSharePoint			•		•		•
DelimitedList			•		•		•
Encode			•		•		•
Encrypt			•		•		•
ExecuteAction							
ExecuteSqlQuery				•	•		
ExportView		•					
FilterOwssvr			•		•		
FormatNumber			•		•		•
GenerateGuid			•		•		•
GetFileName			•		•		
GetFormProperty			•		•		•
GetInputParameter			•		•		•
GetListGuid			•		•		•
GetTermsByLabel			•		•		
GetTermPath			•		•		
GetTermSetIds							
GetTermSets			•		•		



Commands	Filler Only	Full trust	Sandbox possible	Admin Deploy	Browser compatible	DBXL required	FormsViewer compatible
GetUserProfileByName			•		•		•
GetWeekDay			•		•		•
GetXml			•		•		•
Insert			•		•		•
InsertPi			•		•		•
IsDestReachable			•		•		
IsDirty			•		•		
JsonToXml			•		•		•
LoadResource							•
ListDirectory	•	•					
MakeRequest		•		•	•		•
ModifySqlQuery				•	•		
OpenForm	•						
OpenUrlInBrowser	•	•					
Print	•	•					
RefreshSharePointListItems			•		•		•
RemoveDbxIPi			•		•	•	•
RemovePi			•		•		•
RenderForm		•		•	•		
ReplaceString			•		•		•
ResetDbxlVersion			•		•	•	
SaveToDbxl			•		•	•	•
SaveToPath	•	•					
SaveToSharePoint			•		•		•
SetAutoUpdate			•				
SetCaption	•						
SetCase			•		•		•
SetDefaultView			•		•		•
SetDirty	•				•		
SetFocus	•				•		
SetSave	•						
SetValue			•		•		•
SetXml			•		•		•
SortTable			•		•		•
SubmitAsUser		•		•	•		



Commands	Filler Only	Full trust	Sandbox possible	Admin Deploy	Browser compatible	DBXL required	FormsViewer compatible
SubmitToDbxl			•		•	•	•
SubmitToSharePoint			•		•		•
SubmitToSharePointList			•		•		•
SwapDomWithDocument			•		•	•	•
Transform			•		•		•
ValidateUser	•	•					
XmlToJson			•		•		•



COMMAND INDEX

A

Abs, 29
 Acos, 29
 AlterImage, 13
 Asin, 30
 AssignDocument, 10
 Atan, 30
 Atan2, 30
 AttachToSharePointList, 44

B

BigMul, 30

C

Ceiling, 31
 ChangeConnectionUrl, 3
 ChangeSubmitUrl, 3
 ClearErrors, 20
 CompareStrings, 54
 ConvertTimeZone, 6
 Copy, 37
 CopyRichText, 37
 CopyTable, 38
 Cos, 31
 Cosh, 31
 CreateDirectory, 13
 CreateSharePointFolder, 45

D

DateAdd, 7
 DateDiff, 7
 Ddb, 15
 DecodeBase64, 14
 Decrypt, 55
 Delete, 39
 DeleteFile, 14
 DeleteFromSharePoint, 46
 DelimitedList, 54
 DivRem, 31

E

Encode, 54

EncodeBase64, 55
 Encrypt, 55
 ExecuteAction, 20, 22
 ExecuteSqlQuery, 4
 Exp, 32
 ExportView, 21

F

FilterOwssvr, 46
 Floor, 32
 FormatDate, 8
 Fv, 16

G

GenerateGuid, 56
 GenerateRandomNumber, 32, 33
 GetAttachmentAttribute, 14
 GetDayOfYear, 8
 GetEnvironment, 21
 GetErrorCount, 21
 GetErrors, 22
 GetFileName, 22
 GetFullDateTime, 9
 GetIndex, 57
 GetInputParameter, 23
 GetListGuid, 47
 GetTermPath, 47
 GetTermsByLabel, 47
 GetTermSetIds, 47
 GetTermSets, 48
 GetTimeZone, 9
 GetUserProfileByName, 48
 GetValue, 39
 GetWeekDay, 9
 GetXml, 39

I

Insert, 39, 40
 InsertPi, 23
 Ipmt, 16
 Irr, 17
 IsDestReachable, 5
 IsDirty, 23
 IsNew, 24



IsReadOnly, 24

J

JsonToXml, 35

L

ListDirectory, 14

LoadResource, 35

Log, 33

LookupDns, 5

M

MakeRequest, 36

Mirr, 17

ModifySqlQuery, 6

Move, 40

MoveTable, 40

N

Nper, 17

Npv, 18

O

OpenForm, 24

OpenUrlInBrowser, 25

P

PadString, 57

Pmt, 18

Pow, 33

Ppmt, 18

Print, 25

Pv, 19

Q

QueryData, 6

R

Rate, 19

RefreshSharePointListItems, 49

Relink, 25

RemoveDbxIPi, 10

RemoveFromString, 57

RemovePi, 26

RenderForm, 26

Replace, 41

ReplaceString, 58

ResetDbxlVersion, 10

Round, 33

S

SaveToDbxl, 11

SaveToPath, 26

SaveToSharePoint, 50

SendEmail, 36

SetAutoUpdate, 15

SetCaption, 27

SetCase, 58

SetDefaultView, 27

SetDirty, 27

SetFocus, 27

SetSave, 27

SetValue, 42

SetXml, 28

ShowDialog, 28

Sin, 34

Sinh, 34

Sln, 20

SortTable, 42

Sqrt, 34

SubmitAsUser, 51

SubmitToDbxl, 11

SubmitToSharePoint, 52

SubmitToSharePointList, 52

SwapDomWithDocument, 12

Syd, 20

T

Tan, 34

Tanh, 35

Transform, 43

TrimString, 59

V

ValidateUser, 29

X

XmlToJson, 37



SUPPORT

If you have questions about the information in this document, please contact Qdabra Software for assistance. The qRules product page on Qdabra.com will contain an updated list of all available documentation.

Licensed customers can contact us via Support@Qdabra.com. You can also use the InfoPathDev.com Qdabra Product support forums to request help from the community.

Contact Information

Qdabra Software
218 Main Street, Suite 731, Kirkland, WA 98033

Phone: 877.544.2389
Email: Support@Qdabra.com
Website: <http://www.qdabra.com>
Community: <http://www.infopathdev.com>

