Source URL: [Comprehensive list of Functions in NITRO Studio - Crow Canyon Software Support](https://www.crowcanyon.help/article/comprehensive-list-of-functions-in-nitro-studio/)

Comprehensive list of Functions in NITRO Studio

4062 views  15 min , 2 sec read [2](https://www.crowcanyon.help/article/comprehensive-list-of-functions-in-nitro-studio/)

Applies To: SharePoint On-premises (2013/2016/2019/SharePoint Server Subscription Edition) and SharePoint Online

Description:

This article and related articles describe the syntax and examples of the functions that are available in NITRO Custom Actions and NITRO Workflow components.

**Related articles:**

|  |  |
| --- | --- |
| **S. No**. | **Article** **Link** |
| 1 | [Function: GetGroupMembers and Join](https://www.crowcanyon.help/article/706/) |
| 2 | [Function: Get Column Value For Version](https://www.crowcanyon.help/article/708/) |
| 3 | [Comprehensive List Of Functions In NITRO Studio – 2](https://www.crowcanyon.help/article/710/) |
| 4 | [Function: Format Value 2](https://www.crowcanyon.help/article/713/) |
| 5 | [Function: Append in NITRO Workflows and Custom Actions](https://www.crowcanyon.help/article/712/) |

Functions available in both NITRO Custom Actions and NITRO Workflows:

$add(DateTimeField,TimeSpan)

This function performs the sum of two values or adds a given time span to a date time object.

Syntax:

Column Value: $add(ColumnValue,ColumnValue)

Column name: $add(([ColumnDisplayName|ColumnInternalName]),Days:Hours:Minutes:Seconds)

Query List: $add(QuerylistActionName##ColumnInternalName,ColumnValue)

Variable: $add(WFActionName##ColumnInternalName,ColumnValue)

Example

Column Value – $add(97, 3)

*Output: 100*

Column name – $add([Due Date|DueDate],2:0:0:0)

Due Date: 12/2/20

*Output: 12/4/20*

Query List – $add(GetPurchaseItem##Price,100)

GetPurchaseItem##Price: $1000

*Output: $1100*

WF Variable – $add(WFaction##Price,[TaxAmount|TaxAmount])

WFaction##Price: $1000

Tax Amount: $80

*Output: $1080*

Note: Source and Target columns should be of type Number, currency or Date Time.

$addmonths(DateTimeField,Number)

This function adds a given number of months to a date time.

Syntax

$addmonths([ColumnDisplayName|ColumnInternalName],Value)

Example

1. $addmonths([Due Date|DueDate],5)

Due Date: 12/2/20

*Output: 5/2/21*

* $addmonths([Due Date|DueDate],-5)

Due Date: 12/2/20

*Output: 7/2/20*

Note: Target field should be a Date Time column.

$calcbusinesstime(DateTimeField,TimeSpan)

This function adds business time to a given date time. Business time configuration is read from site regional settings. Go to site settings -> site administration -> Regional settings to check the existing business time and days setup for the site.

Syntax

Column Name: $calcbusinesstime([ColumnDisplayName|ColumnInternalName],Days:Hours:Minutes:Seconds)

Example

Created: 12/2/20

$calcbusinesstime([Created|Created],5:0:0:0)

*Output: 12/9/20*

Note: Target column should be a Date Time column

$calcdate(DateTimeField,TimeSpan)

This function adds specified period to a date time object.

Syntax

$calcdate([ColumnDisplayName|ColumnInternalName],number, Days/Weeks/Months/Years)

Example

Due date: 12/2/20

$calcdate([Due Date|DueDate],5,Months)

*Output: 5/2/21*

$calcdate([Due Date|DueDate],-5,Days)

*Output: 07/2/20*

Note: Works with only Date Time columns.

$divide([value/placeholder],[value/placeholder])

This function divides the first value by the second value. The arguments can be either direct values or placeholders of column values from the SharePoint list item.

Syntax

Column Value:  $divide(ColumnValue,ColumnValue)

Column name:  $divide(([ColumnDisplayName|ColumnInternalName]), ([ColumnDisplayName|ColumnInternalName]))

Query List:  $divide (QuerylistActionName##ColumnInternalName,ColumnValue)

Variable:  $divide (WFActionName##ColumnInternalName,ColumnValue)

Examples

$divide(100/25)

*Output: 4*

$divide([Total Cost|TotalCost],3)

Total Cost: 30

*Output: 10*

$divide(Var1##value,Var2##value)

Var1##value: 30

Var2##value: 3

*Output: 10*

$firstitem([ID|ID])

This function is used in Query List actions to capture the first item ID in an item collection and store this in a variable. This allows for running iterative loops through queried items to get beyond the 100-item query limit. Used in conjunction with $lastitem([ID|ID]) and variables.

Syntax

QueryListName##$firstitem([ID|ID])

$formatvalue(placeholder)

This function returns the given lookup or user column display value. This strips out the leading information that SharePoint adds to Person or Group columns, e.g. “22;#James Restivo”

Syntax

$formatvalue([Placeholder])

Example

Requester: 22;#James Restivo

$formatvalue([Requester|Requester])

*Output: James Restivo*

Supported Columns in Parameters: lookup, person or group. More than two parameters can be used in this function.

Note: Target column type should be string.

$jsonpath2([Value/placeholder],JsonPath)

This function is similar to $jsonpath() defined further below in the documentation. $jsonpath gets the output as array of objects whereas $jsonpath2() directly gives us the element from the array.

First Argument possible parameters:

1. Literal/Custom Value:
   1. Syntax: $jsonpath2(any valid json object,@jsonpath)
2. Other column value (Place holder of current item)
   1. Syntax: $jsonpath2([placeholder],@jsonpath)
3. Other column value (Place holder of query list action executed prior to this)
   1. Syntax: $jsonpath2(QuerylistActionName##Field placeholder,@jsonpath)
4. Variable value
   1. Syntax: $jsonpath2(VariableName##Value,@jsonpath)

**Detailed Examples with output**

Example

My JSON object input for this example:

{“MovieDatabase”: {

                                “movie”: [{

                                                “name”: “The Change-Up”,

                                                “genre”: “comedy”,

                                                “director”: “David Dobkin”,

                                                “Facebook\_like”: 252

                                },

                                {

                                                “name”: “Rise of the Planet of the Apes”,

                                                “genre”: “SciFi”,

                                                “director”: “Rupert Wyatt”,

                                                “Facebook\_like”: 472

                                }]

                }

}

Expression: $jsonpath2(JSON object, @$.MovieDatabase.movie[\*].director)

Output: David Dobkin,Rupert Wyatt

$lastitem([ID|ID])

This function is used in Query List actions to capture the last item ID in an item collection and store this in a variable. This allows for running iterative loops through queried items to get beyond the 100-item query limit. Used in conjunction with $firstitem([ID|ID]) and variables.

Syntax

QueryListName##$lastitem([ID|ID])

$lookupid(placeholder)

This function returns the given lookup/user column’s ID. This is used to compare ID values of lookup items. For instance, by comparing a lookup value on one item to the lookup value on another item you can verify that it is the same value.

Syntax

$lookupid([ColumnDisplayName|ColumnInternalName])

Example

Requester: #22;James Restivo

$lookupid([Requester|Requester])

*Output: 22*

$parselookupvalue(lookup column value)

This function is used in column mappings (create/update item) to set the value for a lookup column.

The function will try to get the item from lookup list with the text value given as parameter and returns the first item matching that value.

The output is the same as the argument you add to the function, however, this will allow you to add a string of text to set the lookup column. Otherwise, you’ll need to add in the leading information that SharePoint adds. For instance, to set a Category value you could directly use “#1;Hardware Problem” in the column mapping. With this function, though, you can use “Hardware Problem” in the argument to get the same thing.

Syntax:

Column Value: $parselookupvalue(LookupColumnValue)

Column name: $parselookupvalue([ColumnDisplayName|ColumnInternalName])

Query List: $parselookupvalue(QuerylistActionName##ColumnInternalName)

Variable: $parselookupvalue(VarCCS1##value)

Example

Column value – $parselookupvalue(Calendar)

Column name – $parselookupvalue([Title|Title])

Query list placeholder – “$parselookupvalue(ConfigurationItem##Title)”

Variable placeholder – “$parselookupvalue(UsersVariable##Value)”

Note: This function is only used in field mappings and the target column should be a lookup column.

$parsemetadatavalue()

This function is used in column mappings (create/update item) to set the value for a Managed Metadata column.

The function will try to set value for Managed Metadata column with the text value given as parameter and returns the first item matching that value.

Syntax:

Column Value: $ parsemetadatavalue(MetadataValue)

Column Name: $ parsemetadatavalue([ColumnDisplayName|ColumnInternalName])

Query List: $ parsemetadatavalue(QuerylistActionName##ColumnInternalName)

Variable: $ parsemetadatavalue(VariableName##value)

Example

$ parsemetadatavalue(Calendar)

Output: Calendar

Note: This function is only used in field mappings and the target column should be a Managed Metadata column.

$parseuservalue(value)

This function is used in column mappings (create/update item) to set the value for a user column.

The function will try to ensure user from web i.e, we will try to resolve the user based on the parameter provided in the function.

Syntax

$parseuservalue(User Display Name/Alias/Email Address)

Example

$parseuservalue([james@contoso.com](mailto:james@crowcanyon365.com))

Output: James Restivo

Note: This function can be used only in field mappings and the target field should be User field.

$product([value/placeholder],[value/placeholder])

This function performs the product of (multiplies) two values.

Syntax:

Column Value:  $product(ColumnValue,ColumnValue)

Column name:  $product(([ColumnDisplayName|ColumnInternalName]), ([ColumnDisplayName|ColumnInternalName]))

Query List:  $product(QuerylistActionName##ColumnInternalName,ColumnValue)

Variable:  $product(VariableName##value,ColumnValue)

Example

Column Value – $product (25, 5)

*Output: 125*

Column name – $product([Price|Price],[Quantity|Quantity])

Price: $1200

Quantity: 4

*Output: $4800*

Query List – $product(GetPurchaseItem##Price,2)

GetPurchaseItem##Price: $1200

*Output: $2400*

Variable – $product(VarCCS1##value,5)

VarCCS1: $500

*Output: $2500*

Note: The values used in the function should be number values or should be resolved to numbers and target column should be number or currency field

$querylistactionname##$count(placeholder)

This function performs the count operation on values returned from parameters.

This can only be used on query list action to count the values from multiple items.

Syntax

QuerylistActionName##$count([ColumnDisplayName|ColumnInternalName])

Example

GetPurchaseItem##$count([Cost|Cost])

$regextract([value/placeholder],@expression)

This function is used to extract a pattern value from a string using regex expression.

Syntax

$regextract([ColumnDisplayName|ColumnInternalName],@expression)

Example

$regextract([Subject|Subject],@.\*\[CaseId: (.+)\].\*)

Subject: Printer broken [CaseId: 2]

*Output: 2*

$split(separator,[value/placeholder])

This function splits the string to a collection of strings using the separator.

Syntax

$split(separator,[ColumnDisplayName|ColumnInternalName])

Example

$split(:,[Title|Title])

Input:

Title: Re: Custom Actions NITRO Studio

Output: [“Re”,”Custom Actions NITRO Studio”]​

$split(@,[Requester|Requester])

Input:

Requester: [james.restivo@contoso.com](mailto:james.restivo@crowcanyon.com)

Output: [“james.restivo”, “contoso.com”]

$strcat(TextField,TextFieldsOrValues)

This function performs concatenation on given parameter values.

Syntax:

Column Value: $strcat(ColumnValue,ColumnValue)

Column name: $strcat((ColumnDisplayName | ColumnInternalName),ColumnValue/([ColumnDisplayName|ColumnInternalName]))

Query List: $strcat(QuerylistActionName##ColumnInternalName,ColumnValue)

Variable: $strcat(WFActionName##ColumnInternalName,ColumnValue)

Example

Column Value – $strcat(Crow Canyon ,rocks)

*Output: Crow Canyon rocks*

Column name – $strcat([Title|Title], new)

Title: Request for PTO

*Output: Request for PTO new*

Query List – $strcat(GetPurchaseItem##Price,100)

WF Variable – $strcat(WFaction##Title,[Title|Title])

Note: Target column should be a text or Multi Line text column. This function accepts more than two parameters.

$strlen(value)

This function returns the length of the specified parameter return value.

Syntax:

Column Value: $strlen(LookupColumnValue)

Column name: $strlen([ColumnDisplayName|ColumnInternalName])

Query List: $strlen(QuerylistActionName##ColumnInternalName)

Variable: $strlen(WFActionName##ColumnInternalName)

Example

Column value – $strlen(crowcanyon)

*Output: 9*

Column name – $strlen([Title|Title])

Query list place holder – “$strlen(ConfigurationItem##Title)”

WF variable place holder – “$strlen(UsersVariable##Value)”

Note: Target column should be number/currency/text.

$subtract(NumberField,Number)

This function is used to subtract two values or subtract a given time span from a date time object.

Syntax:

Column Value: $subtract(ColumnValue,ColumnValue)

Column name: $subtract (([ColumnDisplayName|ColumnInternalName]),Days:Hours:Minutes:Seconds)

Query List: $subtract(QuerylistActionName##ColumnInternalName,ColumnValue)

Variable: $subtract(WFActionName##ColumnInternalName,ColumnValue)

Example

Column Value – $subtract(103, 3)

*Output: 100*

Column name – $subtract([Due Date|DueDate],2:0:0:0)

Due Date: 12/2/20

*Output: 11/30/20*

Query List – $subtract(GetPurchaseItem##Price,100)

WF Variable – $subtract(WFaction##Price,[Discount|Discount])

Note: Source and Target columns should be either Number or Currency or Date Time columns.

Today

$usercollection([value/placeholder],[value/placeholder])

This function gets the collection of users from different fields that can be mapped to a multi user column.

When configured in a WF Variable action, the collection of users returned can be used to create an item for each user.

Syntax

$usercollection([ColumnDisplayName|ColumnInternalName],[ColumnDisplayName|ColumnInternalName]).

Example

Create Task for each user entered in ‘Level 1 Technician’, ‘Level 2 Technician’.

$usercollection([Level 1 Technician|Level1Technician],[Level 2 Technician|Level 2 Technician]).

Input:

Level 1 Technician: Pavan Kumar, Prakash Arya

Level 2 Technician: James Restivo, Scott Restivo

Output:

Create Task 1 for Pavan Kumar

Create Task 2 for Prakash Arya

Create Task 3 for James Restivo

Create Task 4 for Scott Restivo

For more detailed use case, please refer [Using User Collection](https://www.crowcanyon.info/nitro/appmanual_v2/index.html?use-case---using-usercollectio.html).

Note

Source column should be a person or group column.

$xpath([value/placeholder],@xpath)

This function is used to navigate through elements and attributes in an XML.

Syntax

Column Name: $xpath([value/placeholder],xpath)

Query List: $xpath(QuerylistActionName##ColumnInternalName,xpath)

WF Variable: $xpath(WFActionName##ColumnInternalName,xpath)

Example

$xpath([xmlvalue|xmlvalue], /birds/parrot)

Input: Value in ‘xmlvalue’ column is <birds><parrot><color>green</color></parrot><pigeon><color>white</color></pigeon></birds>

Output: <parrot><color>green</color></parrot>

$xpath(TicketItem##Xml,/Category/CategoryOwner)

Input: TicketItem is a query list that is getting the XML value from Category or a lookup list item.

Output: Pavan Kumar

Where Category Owner of the selected category is Pavan Kumar

Note: To repeat attribute/element collection on a workflow action, this function should be configured in WF Variable action and can be used later in field mappings of create item action by using the syntax ‘@{CollectionValue}tag/attribute name’.

For single value, directly use full function.

Functions exclusively available in NITRO Workflows

$datediffbusinesshours(DateTimeField,DateTimeField)

This function is used to get the business hours difference of two Date and Time fields. Site Regional Time zone settings will be considered in this case to calculate the business hours

Syntax

$datediffbusinesshours(DateTimeField,DateTimeField)

Example

$datediffbusinesshours([Start Time|StartTime],[End Time|EndTime])

Output: 12

Note: Target columns should be either single line of Text/Number column

$datediffhours(DateTimeField,DateTimeField)

This function is used to get the total hours difference of two Date and Time fields.

Syntax

$datediffhours(DateTimeField,DateTimeField)

Example

$datediffhours([Start Time|StartTime],[End Time|EndTime])

Output: 18

$extractheaders(Separator,[value/placeholder],option)

This function gets a collection of key value pairs from a value based on separator. It can only be used in WF Variable action.

Syntax

$extractheaders(separator,[ColumnDisplayName|ColumnInternalName],firstupper/upper/lower )

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter Sequence | Description | Optional / Default Value | Example |
| 1 | Separator between name and value. Also, it is the separator between the value and next key | Required | Semicolon ;   Note: Comma cannot be used as a separator as it is reserved character in function syntax |
| 2 | Column specifier in format [ColumnDisplayName|ColumnInternalName] | Required | We can specify column name that has the text that needs to be parsed to extract name value pairs |
| 3 | Optional Parameter | Optional | firstupper: to extract the key value with setting first character as capital letter upper: to extract the key value with setting all characters as capital letters lower: to extract the key value with setting all characters as small letters |

Example

Find the ‘Importance’ of the email received by SharePoint announcement list (Incoming email enabled list)

WF Variable:

WfVar: $extractheaders(:,[E-Mail Headers|EmailHeaders],firstupper)

Input:

x-sender: James@contoso.com

x-receiver: contoso@server.domain.com

Sensitivity: NORMAL

Importance: Normal

Received: from …

….

In Field Mappings: WfVar##Importance

Output: Normal

Note: Comma can’t be used like separator as it is used as function parameters separator.

$getaduserproperty

$getexpressionvalue

$getspgroupmembers(displayValue,joinchar,UserField)

This function is used to get the list of users that are present in a SharePoint group.

Syntax

$getspgroupmembers(displayValue,joinchar,UserColumn)

Example

$getspgroupmembers(name,;,[Assigned Team|AssignedTeam])

Input: ‘Assigned Team’ is ‘Crow Canyon Service Request Owners’

Output: James Restivo;Crow Admin;Pavan Kumar;

Note: Target column need to be either Single line of text/multiple lines of text.

$getspuserproperty(displayValue,joinchar,UserField)

This function is used to get the values of Person or Group column and set the value to a single line of text column.

Syntax

$getspuserproperty(displayValue,joinchar,UserColumn)

displayValue can be email, name, id, loginname, Userprincipalname and manager

joinchar is separator like ;

UserField is a person or group column placeholder

Example

$getspuserproperty(email,;,[Assigned Staff|AssignedStaff])

Input: Pavan Kumar, James Restivo

Output: pavan@contoso.com;james@contoso.com

Note: Target column should be either Single line of text or multiple lines of text field.

$getspuserproperty(manager,;,[Requester|Requester])

Input: James Restivo

Output: Scott Restivo

$itemsharedwith(Id)

This function is used to get the list of users for which current Item is shared with or has access to view the item.

Syntax

$itemsharedwith(Id)

Example

Get all users and insert in a multiple selection enabled person or group column ‘All Techinicians’

$itemsharedwith([ID|ID])

Input: [ID|ID] (list item ID)

Output: Pavan Kumar;James Restivo;Scott Restivo

Note: Target column need to be a multiple selection enabled Person or group type.

{ListId}

This function returns the List ID of the list on which workflow is configured on and can be set in text/note columns

Syntax

{ListId}

{ListName}

This function returns the Title of the list on which workflow is configured on and can be set in text/note columns

Syntax

{ListName}

Now

$regexreplace(,,)

The function $regexreplace(,,) has the feature to check for a matching value based on Regular Expression and replace it with another string in a column.

Syntax

 $regexreplace(Column Placeholder,Regular expression,replace value,type)

Example

Remove spaces in given text

$regexreplace([Title|Title],@\s,,,1)

Input: Title has value as “This is a Test”.

Output: ThisisaTest

{SiteCollectionId}

This function returns the Site Collection GUID data and can be set in text/note columns

Syntax

{SiteCollectionId}

{SiteCollectionUrl}

This function returns the Site Collection URL and can be set in text/note columns

Syntax

{SiteCollectionUrl}

$Sum(NumberField,NumberField,number)

This function will sum up several number fields and/or numbers

Syntax

$sum([Column1 Display Name|Column1InternalName],[Column2 Display Name|Column2InternalName])

Example

$sum([Total Cost|TotalCost],[Tax|Tax])

UtcNow

$variablecollection(Separator,[value/placeholder],option)

This function gets the collection of values from parameters and combines them into a single value by joining the collection of values with a separator. It can also be used to get the collection of column values from query list items.

Syntax

SharePoint Online: $variablecollection(separator,[ColumnDisplayName|ColumnInternalName])

SharePoint On-Premises: $variablecollection([ColumnDisplayName|ColumnInternalName],separator)

Example

Combine Requester and Assigned To column values and insert in a multiple person or group column: $variablecollection(;#,[Requester|Requester],[AssignedTo|AssignedTo])

Input:

Requester: Pavan Kumar

Assigned To: James Restivo

Output:

Pavan Kumar;#James Restivo

Query list Syntax

$querylistactionname##$variablecollection(separator,[value/placeholder],optional)

Example 1

Read related child tasks assignee and insert in a multiple selection person or group column on a parent item.

$GetRelatedTasks##$variablecollection(;#,[Assigned To|AssignedTo])

Input:

Task 1 Assignee: Pavan Kumar

Task 2 Assignee: James Restivo

Output:

Pavan Kumar, James Restivo

Note: More than two parameters can be used in this function

Example 2

When we have title with | as separator, and we need to use them in create item action each item separated by |, then we can define this variable collection as variable.

Input:

Title: Email | Calendar | Outlook

We can use this Wf variable of variable collection to create each item for each collection value with @{CollectionValue} as column mapping for each task when inserted in Title column mapping

Output:

Task 1 created with Email as Title

Task 2 created with Calendar as Title

Task 3 created with Outlook as Title

$variablecollection2(Separator,unique,[Placeholder1],[Placeholder2])

This function is similar to $variablecollection(), but the advantage of this function is to get multiple column values from each item and place in a multiple lines of text column.

Syntax

SharePoint Online: $variablecollection(separator,[ColumnDisplayName|ColumnInternalName])

SharePoint On-Premises: $variablecollection([ColumnDisplayName|ColumnInternalName],separator)

Example

$variablecollection2(;#,[Requester|Requester],[AssignedTo|AssignedTo])

Query list

$querylistactionname##$variablecollection2(separator,unique,[value/placeholder], [value/placeholder], [value/placeholder])

Note: More than two parameters and multiple column placeholders can be used in this function.

{WebId}

This function returns the Current web GUID and can be set in text/note columns

Syntax

{WebId}

{WebUrl}

This function returns the Current web URL and can be set in text/note columns

Syntax

{WebUrl}

Functions exclusively available in NITRO Custom Actions

$getuserattr(User Id/{Me}/[UserColumn],Email/DisplayName)

This function is used to return Email/Display Name attribute of a user based on User ID or Logged in user or a person or group column.

Syntax

$getuserattr(User Id/{Me}/[UserColumn],Email/DisplayName)

Examples:

1. Input: $getuserattr(15,Email)

Output: [pavan@contoso.com](mailto:pavan@contoso.com)

Where ‘Pavan Kumar’ User id is 15

* $getuserattr([Created By|Author],Email)

Output: [jamesr@contoso.com](mailto:jamesr@contoso.com)

Where item is created by ‘James Restivo’

* $getuserattr({Me},DisplayName)

Output: Pavan Kumar

Where logged in user is ‘Pavan Kumar’

$jsonpath([value/placeholder],@jsonpath)

This function is used to navigate through objects in an JSON object/array. This function takes two arguments as input. First argument is one of the following possible parameters and returns the value as per the “json path” given in second argument. Second argument is always expected to be a valid json path.

First Argument possible parameters:

1. Literal/Custom Value:
   1. Syntax: $jsonpath (any valid json object,@jsonpath)
2. Other column value (Place holder of current item)
   1. Syntax: $jsonpath ([placeholder],@jsonpath)
3. Other column value (Place holder of query list action executed prior to this)
   1. Syntax: $jsonpath (QuerylistActionName##Field placeholder,@jsonpath)
4. Variable value
   1. Syntax: $jsonpath (VariableName##Value,@jsonpath)

**Detailed Examples with output**

Example 1

My JSON object input for this example:

{“MovieDatabase”: {

                                “movie”: [{

                                                “name”: “The Change-Up”,

                                                “genre”: “comedy”,

                                                “director”: “David Dobkin”,

                                                “Facebook\_like”: 252

                                },

                                {

                                                “name”: “Rise of the Planet of the Apes”,

                                                “genre”: “SciFi”,

                                                “director”: “Rupert Wyatt”,

                                                “Facebook\_like”: 472

                                }]

                }

}

Expression: $jsonpath(JSON object, @$.MovieDatabase.movie[\*].director)

Output: [“David Dobkin”,”Rupert Wyatt”]

Example 2

My input is coming from a column of Query List Action (action name: TicketItem, Cities is a column name in the query list action list) result list item. And sample input is same as above,

Expression 1: $jsonpath(TicketItem##Movies, @$.MovieDatabase.\*)

Output: [[{“name”:”The Change-Up”,”genre”:”comedy”,”director”:”David Dobkin”,”Facebook\_like”:252},{“name”:”Rise of the Planet of the Apes”,”genre”:”SciFi”,”director”:”Rupert Wyatt”,”Facebook\_like”:472}]]

Expression 2: $jsonpath(TicketItem##Movies, @$.MovieDatabase..Facebook\_like)

Output: [252,472]

Expression 3: $jsonpath(TicketItem##Movies, @$..movie[(@.length-1)])

Output: [{“name”: “Rise of the Planet of the Apes”,”genre”: “SciFi”,”director”: “Rupert Wyatt”,”Facebook\_like”: 472}]

$parsefieldvalue(placeholder,Id/Value Type,Separator)

This function is used to return required output from field value object

Default value type: **String**

Default Separator: **; (semi colon)**

$parsefieldvalue(Column placeholder)

$parsefieldvalue(Column placeholder, value type)

$parsefieldvalue(Column placeholder, value type, separator)

Available value types:

* **Id**
* **Value**
* **Email**
* **Dateonly**
* **url**
* **string**
* **object**

Syntax

$parsefieldvalue(Column placeholder)

Example:

1. $parsefieldvalue([Requester|Requester],Email,;)

Input: Requester, a person or group column value is James Restivo

Output: [james@contoso.com](mailto:james@contoso.com)

* $parsefieldvalue([Category|Category],Id,;)

Input: Category is a lookup column that has value ‘Networking’

Output: 13  
where 13 is the item id of the list item ‘Networking’ in lookup list