# QPR Web Services Extensions Expression Language

**Version 1.6.0 (28.2.2016)**

## Contents

# 1 Overview

This document describes an expression language used in several QPR Suite accelerators and extensions. Expressions (formulas) make it possible to define values dynamically, i.e. values are calculated at runtime.

The evaluation process is case sensitive meaning parameter and function names must be written with right case letter. (Actually all function names are not case sensitive, but it’s easiest to consider they are).

Expressions can be combined using **operators**. Expression's priority order starting from largest priority is: primary, unary, power, multiplicative, additive, relational, logical **and**, logical **or**. Following operators can be used:

- Logical or: **or**, **||** (these are equivalent)
- Logical and: **and**, **&&** (these are equivalent)
- Relational: **=**, **!=**, **<>**, **<**, **<=**, **>**, **>=** (!= and <> are equivalent)
- Basic calculations: **+**, **-**, *, **/**, **%**
- Bitwise: **&** (bitwise and), | (bitwise or), **^** (bitwise xor), **<<** (left shift), **>>** (right shift)
- Unary: **!**, **not**, **-**, **~** (bitwise not)
- functions: **Abs(1)**, **doSomething(1, 'dummy')**

In expressions, characters **\** and **'** need to be written as **\\** and **\\'** (i.e. escaped).

The expression language supports any .Net 4.5 datatype. Following table contains examples, how to write literals of basic datatypes.

| Data type | Example |
|---|---|
| integer (int) | 123456 |
| double | 123.456, 0.123 |
| boolean | true, false |
| string | 'Hello world!' |

When operating with dates, all dates returned by QPR Web Services are strings formatted in XML date format (yyyy-MM-ddTHH:mm:ss.fffzzz). They can be converted to datetimes using function **StringToDate**.

In the expressions any unicode characters can be defined using syntax **\uxxxx**. For example, new line is **\u000D\u000A** (carriage return and line feed characters used in Windows systems).

# 2 Arithmetic functions

| Function | Description | Example expression | Result |
|---|---|---|---|
| Abs | Returns the absolute value of a specified number. | Abs(-1) | 1 |
| Acos | Returns the angle whose cosine is the specified number. | Acos(1) | 0 (double) |
| Asin | Returns the angle whose sine is the specified number. | Asin(0) | 0 (double) |
| Atan | Returns the angle whose tangent is the specified number. | Atan(0) | 0 (double) |

| | | | |
|---|---|---|---|
| Ceiling | Returns the smallest integer greater than or equal to the specified number. | Ceiling(1.5) | 2 (double) |
| Cos | Returns the cosine of the specified angle. | Cos(0) | 1 (double) |
| Exp | Returns e raised to the specified power. | Exp(0) | 1 (double) |
| Floor | Returns the largest integer less than or equal to the specified number. | Floor(1.5) | 1 (double) |
| IEEERemainder | Returns the remainder resulting from the division of a specified number by another specified number. | IEEERemainder(3, 2) | -1 (double) |
| in | Returns whether an element is in a set of values. | in(1 + 1, 1, 2, 3) | true |
| Log | Returns the logarithm of a specified number. | Log(1, 10) | 0 (double) |
| Log10 | Returns the base 10 logarithm of a specified number. | Log10(1) | 0 (double) |
| Max | Returns the larger of two specified numbers. | Max(1, 2) | 2 |
| Min | Returns the smaller of two numbers. | Min(1, 2) | 1 |
| Pow | Returns a specified number raised to the specified power. | Pow(3, 2) | 9 (double) |
| Random | Returns a random value between 0 and 1 (the value can be 0 but cannot be 1). | Random() | 0,358024762 |
| Round | Rounds a value to the nearest integer or specified number of decimal places. | Round(3.222, 2) | 3.22 (double) |
| Sign | Returns a value indicating the sign of a number. | Sign(-10) | -1 |
| Sin | Returns the sine of the specified angle. | Sin(0) | 0 (double) |
| Sqrt | Returns the square root of a specified number. | Sqrt(4) | 2 (double) |
| Tan | Returns the tangent of the specified angle. | Tan(0) | 0 (double) |
| Truncate | Calculates the integral part of a number. | Truncate(1.7) | 1 |

# 3 String functions

| Function | Parameters | Description |
|---|---|---|
| CharAt (string) | - input string<br>- index number (int) | Return character of the **index number** position in the **input string**. The indexing starts from zero. |
| Contains (boolean) | - first string<br>- second string | Return true if the **first string** contains the **second string**; otherwise false. |
| EndsWith (boolean) | - first string<br>- second string | Return true if the **first string** ends with the **second string**; otherwise false. |
| IndexOf (int) | - first string<br>- second string<br>- start index (int) | Return the index number of the first occurrence of the **second string** in the **first string**. Indexing starts from zero. If the **start index** is provided, the search is started from that index. |
| LastIndexOf (int) | - first string | Return the index number of the last occurrence of the **second string** in the **first string**. Indexing starts |

| | | |
|---|---|---|
| | - second string<br>- index (int) | from zero. If the **start index** is provided, the search is started from that index (calculated from the start of the string) towards the beginning of the string. |
| Length (int) | - input string | Return the number of characters in the **input string**. |
| RemoveIllegalChars | - input string<br>- allowed characters (string)<br>- replace string | Removes from the input string all characters that are not part of the listed chars. If the replace string is provided, removed characters are replaces with that one. Examples:<br>**RemoveIllegalChars('ABCDE', 'BD', '')** gives BC<br>**RemoveIllegalChars('ABCDE', 'BD', '_')** gives _B_D_ |
| Replace (string) | - first string<br>- second string<br>- third string | Replaces all occurrencies of the **second string** with the **third string** in the **first string**. Example**: Replace('abcd', 'b', 'e')** gives **aecd**. |
| StartsWith (boolean) | - first string<br>- second string | Return true if the **first string** starts with the **second string**; otherwise false. |
| Substring (string) | - input string<br>- start index (int)<br>- length (int) | Returns a substring of the **input string** starting from the **start index**. If the **length** is provided, the returned string contains maximum of that number of characters. |
| ToLower (string) | - input string | Return a string where all the characters of the **input string** have been converted to lower case characters. |
| ToUpper (string) | - input string | Return a string where all the characters in the input string have been converted to upper case characters. |
| Trim (string) | - input string<br>- string array | Removes spaces from the start and end of the **input string** (if **string array** is not provided). If the string array is provided, the string in the array are removed from the input array. |

# 4 Datetime functions

| Custom function | Parameters | Description |
|---|---|---|
| AddMilliseconds (datetime) | - datetime<br>- milliseconds (double) | Adds the specified number of milliseconds to a datetime. |
| AddMonths (datetime) | - datetime<br>- number of months (int) | Adds the specified number of months to a datetime. Negative number of months means number of months to subtract. This function takes into account years, so e.g. when adding one month to a date in December, the result is January in the next year. |
| AddTimespan | - datetime<br>- timespan | Adds the specified timespan to a datetime. |
| CompareDates (int) | - datetime 1<br>- datetime 2 | Compares two datetimes and returns zero when the datetimes equals, greated than zero if the former is later, and less than zero if the former is earlier. |

| | | |
|---|---|---|
| CurrentDateTime (datetime) | [none] | Returns a datetime representing the current date and time as UTC. As there are no parameters, use syntax **CurrentDateTime()** |
| FromMetricsDateFormat (datetime) | - numerical value (string, double, int) | Returns a date that this the provided number of days from 1.1.1900 00:00:00 UTC. QPR Metrics date values uses this date format. |
| GetDate (int) | - datetime | See (1) below (property: Date) |
| GetDay (int) | - datetime | See (1) below (property: Day) |
| GetMonth (int) | - datetime | See (1) below (property: Month) |
| GetTicks (int) | - datetime | See (1) below (property: Ticks) |
| GetYear (int) | - datetime | See (1) below (property: Year) |
| NewDatetime | - year (int)<br>- month (int)<br>- day (int)<br>- hour (int)<br>- minute (int)<br>- second (int)<br>- millisecond (int) | Returns a new datetime with the following parameter values. The date is speficied as UTC.<br>- year: 1 - 9999<br>- month: 1 - 12<br>- day: The day 1 through the number of days in month<br>- hour: 0 - 23<br>- minute: 0 - 59<br>- second: 0 - 59<br>- millisecond: 0 - 999 |
| SubtractTimespan | - datetime<br>- timespan | Substracts the specified timespan from the datetime. |
| Timespan (datetime) | - days (int)<br>- hours (int)<br>- minutes (int)<br>- secords (int)<br>- milliseconds (int) | Creates a new timespan. A timespan represent an interval between two datetimes. |
| ToMetricsDateFormat (double) | - datetime | Returns number of days between 1.1.1900 00:00 UTC and the provided datetime. QPR Metrics handles date values using this format. Also the function is needed for criteria in Web Service's QueryObjects, e.g. **'createddate>' + ToMetricsDateFormat(CurrentDateTime())** |

(1) DateTime properties in .Net Framework 4.5: https://msdn.microsoft.com/en-us/library/system.datetime_properties(v=vs.110).aspx

# 5 Array functions

An array is a list of objects of any type, such as strings or integers. Arrays may contains same valued items multiple times.

| **Function** | **Parameters** | **Description** |
|---|---|---|

| | | |
|---|---|---|
| ArrayExcept (array) | - array<br>- array | Returns an array which contains items that are in the first array but not in the second array. |
| ArrayIntersect (array) | - array<br>- array | Returns an array which contains items that are in both provided arrays. |
| ArrayReverse (array) | - array | Reverses an array, i.e. first element becomes last and so on. |
| ArraySize (int) | - array | Calculates number of items in an array. |
| ArraySort (array) | - array to sort<br>- sort order (array) | Sorts an array provided as a first parameter. The second parameter provides an array that contains an explicit order for the items to be used in sorting. The rest of the items that are not listed, are sorted normally and placed in the end of the outputed array. |
| ArrayUnion (array) | - array<br>- array | Joins two arrays. Duplicates are not removed. If only one parameter is provided, that parameter is expected to contain an array, item of which are arrays. In that case all the items of all arrays are joined. |
| ArrayUnique (array) | - array | Returns an array where duplicate values are removed. Order of items is preserved. |
| ArrayWhere (array) | - array<br>- expression (string) | Filters out all items in an array, where the expression is evaluated as false. The expression must return boolean value. Available arguments in the expression:<br>- **item** (item in the array)<br>- **index** (iteration order number starting from 0) |
| Average (double) | - array | Calculates an average of array items. Array must contains numerical data. |
| Concatenate (string) | - array<br>- separator (string) | Concatenates items of an array into a string separated by a provided separator. Array must contain items that can be converted into string. |
| IndexOfAnyInArray (int) | - array<br>- array | Returns an index number of the any of objects of the second array in the first array. The second array objects are searched from left to right. First object array starting from index 0. If the first array doesn't contain any of objects in the second array, -1 is returned. |
| IndexOfInArray (int) | - array<br>- item to search (object) | Returns an index number of an object in the array. First object array starting from index 0. If object is not found, -1 is returned. |
| ItemAt (object) | - array<br>- int | Returns an item from an array in the provided index number. The first item has an index number zero (0). |
| Median (double) | - array | Returns the median of the provided array. |
| Minimum (double) | - array | Returns the smallest number in the provided array. |
| Maximum (double) | - array | Returns the largest number in the provided array. |
| OnlyValue (object) | - array | Returns the only value of the array. Throws an exception if the array contains more than one value. Return null if the array contains no values. |

|  |  |  |
|---|---|---|
| Sum (double) | - array | Calculates a sum of array items. Array must contains numerical data. |
| Transform (array) | - array<br>- expression (string) | Transforms every item of an array to another array using the provided expression. The expression has following arguments:<br>- **value** which gets the value of the item.<br>- **index:** iteration order number starting from 0<br>E.g. **transform([inputarray], 'substring([value], 5)')** returns first five characters from every string of the array. |

## 6 Dictionary functions

| Function | Parameters | Description |
|---|---|---|
| Dictionary (dictionary) | - keys (keys array)<br>- values (object array) | Creates a new dictionary based on two arrays. The two arrays must not be null, and must be the same length. |
| GetDictionaryValue (object) | - dictionary<br>- key (string)<br>- default value (object) | Returns a value from a dictionary based on key. If the key is not found and a default value is defined, the default value is returned. If the key is not found and a default value is not defined, an exception is thrown. |
| ContainsValue (boolean) | - dictionary<br>- key (string) | Returns true if the provided dictionary contains the provided value; otherwise false. |

## 7 Conversion functions

| Function | Parameters | Description |
|---|---|---|
| ArrayToDataset (dataset) | - array<br>- column name (string) | Converts an array to a dataset. The dataset has one column which name is given as a second parameter. |
| ConvertToDouble (double) | - input value (string)<br>- value in error (double) | Converts a string value to a double value. The input value can also be in other formats that can be converted to double. The second parameter is optional, and it's returned if the conversion fails. If it's not defined and the conversion fails, an error is thrown. |
| ConvertToInt (int) | - input value (string)<br>- value in error (int) | Converts a string value to an integer value. The input value can also be in other formats that can be converted to integer. The second parameter is voluntary, and it's returned if the conversion fails. If it's not defined and the conversion fails, an error is thrown. |
| ConvertToString | - input value (object)<br>- date format (string) | Converts a variable of many datatypes to string as follows:<br>- null: empty<br>- DateTime: formatted using provided or XML date format |

| | | |
|---|---|---|
| | - value in error (string) | - Boolean: "true" or "false"<br>- Array: items comma separated enclosed with []<br>The second parameter is optional. The third parameter is optional, and it's returned if the conversion fails. If it's not defined and the conversion fails, an error is thrown. |
| DatasetColumnToArray (array) | - dataset<br>- column name (string) | Converts a column of a dataset to an array. The column is referenced using the column name. |
| DatasetRowToArray (array) | - dataset<br>- row number (int) | Converts a row of a dataset to an array. The row is referenced using the row number (the first is 1). |
| DateToString (string) | - date to convert (datetime)<br>- local time (boolean)<br>- dateformat (string) | Converts a datetime to a string. The local time parameter is optional; if omitted, the string is assumed to represent local time. If the third parameter is omitted, the string is interpreted as XML schema date format. More information of how to construct the dateformat: https://msdn.microsoft.com/en-us/library/8kb3ddd4(v=vs.110).aspx |
| FileDataToString (string) | - file data (byte array) | Converts a file provided as a byte array into string using UTF8 encoding. If null is provided, null is returned. If zero valued array is provided, an empty string is returned. |
| HtmlEncode (string) | - string | Performs an HTML encoding. More information: https://msdn.microsoft.com/en-us/library/system.net.webutility.urlencode(v=vs.110).aspx |
| InstanceIdFromFullId (string) | - element full id (string) | Extracts an instance id from a full id. The function returns an empty string, if the instance id cannot be extracted.<br>Example: **InstanceIdFromFullId('SC.6346904. 5290187.4353690')** returns 4353690 |
| ItemsToArray (array) | - object 1<br>- object 2<br>- … | Converts provided parameters as an array. There must be at least one parameter. Number of parameters is unrestricted. |
| ModelIdFromFullId (string) | - element full id (string) | Extracts a model id from a full id. The function returns an empty string, if the model id cannot be extracted.<br>Example: **ModelIdFromFullId('SC.6346904. 5290187')** returns 6346904 |
| NumberToString (string) | - value (int, double)<br>- format (string)<br>- value in error (string) | Converts a numerical value to string. This function can be used when there are special requirements related to e.g. thousand separators. Behaviour of this function depends on the server's locale settings. If the convertion fails and the value in error is defined, that value is returned. Otherwise an error is thrown. More information about formattings:<br>- https://msdn.microsoft.com/en-us/library/dwhawy9k(v=vs.110).aspx#NFormatString<br>- https://msdn.microsoft.com/en-us/library/0c899ak8(v=vs.110).aspx |
| ObjectIdFromFullId (string) | - element full id (string) | Extracts an object id from a full id. The function returns an empty string, if the object id cannot be extracted. |

| | | |
|---|---|---|
| | | Example: **ObjectIdFromFullId('SC.6346904. 5290187')** returns 5290187 |
| PresentObjectAsString | - input object<br>- date format (string)<br>- value in error (object) | Presents variables of many datatypes as string for debugging porposes as follows:<br>- null: "[null]"<br>- DateTime: formatted using provided or XML date format<br>- String: enclosed using quotes<br>- Boolean: "true" or "false"<br>- DataSet: {{}}<br>- Array: items comma separated enclosed with []<br>The second and third parameters are optional. |
| StringToArray (array) | - string to convert<br>- separator (string)<br>- remove empties (bool) | Converts a string to an array using provided separator string. If remove empties is true, there are no empty strings in the output array. |
| StringToDate (datetime) | - string (string to convert)<br>- local time (boolean)<br>- string (dateformat) | Converts a string to a datetime. The local time parameter is optional; if omitted, the string is assumed to represent local time. If the third parameter is omitted, the string is interpreted as XML schema date format. More information of how to construct the dateformat: https://msdn.microsoft.com/en-us/library/8kb3ddd4(v=vs.110).aspx |
| UrlEncode (string) | - string | Performs a URL encoding. More information: https://msdn.microsoft.com/en-us/library/4fkewx0t(v=vs.110).aspx |

# 8 QPR Web Services functions

| Function | Parameters | Description |
|---|---|---|
| BelongsToGroup (boolean) | - group name (string) | Returns true if the current user belongs to the provided group; otherwise false. QPR Web Services query is made to get the information. The current user is the user account which is logged to the QPR Web Services when the query is made. False is returned if the provided group doesn’t exist. Upper and lower case characters need to be written correctly.<br>Example: **BelongsToGroup("Administrators")** |
| CurrentUser (string) | - attribute name (string) | Returns any attribute of the current user object. If no attribute is provided, user id is returned (attribute "id").<br>Example: **CurrentUser()** |
| GetAttribute (string) | - objectid (string)<br>- attribute (string)<br>- options (string) | QPR Web Service’s GetAttributeAsString operation, see http://kb.qpr.com/qpr2015-1/index.html?getattributeasstring.htm. |

| | | |
|---|---|---|
| GetAttributes (string array) | - objectid (string)<br>- attributes (string)<br>- options (string) | Returns multiple attributes of a single object as an array. Attributes are defined as comma separated. Based on QPR Web Service's QueryObjects operation. |
| GetPortalUrl (string) | - objectid (string)<br>- view (string)<br>- options (string) | QPR Web Service's GetPortalUrl operation, see http://kb.qpr.com/qpr2015-1/index.html?getportalurl.htm. |
| LatestValuePeriod (string) | - objectid (string)<br>- series symbol or series id (string) | Returns the id of the period that contains the latest (newest) measure value. Return an empty string, if no period contains a value. This information can be used e.g. in web service attribute **'prettyvalue(period=' + [periodid] + ')'** |
| QueryObjects (string array) | - query (string)<br>- filter (string)<br>- sortby (string)<br>- attributes (string)<br>- options (string) | Executes QPR Web Service's QueryObjects and returns results as a string array. Only one attribute should be defined for "attributes" parameter. |
| QueryObjectsAverage (double) | - query (string)<br>- filter (string)<br>- attributes (string)<br>- options (string) | Average of queried objects numerical attributes. Only one attribute is defined in "attributes". Error is thrown if attribute values are not numerical. |
| QueryObjectsConcatenate (string) | - query (string)<br>- filter (string)<br>- sortby (string)<br>- attributes (string)<br>- options (string)<br>- separator (string) | Concatenates attribute values of all queried objects using defined separator. Only one attribute is defined in "attributes". |
| QueryObjectsCount (int) | - query (string)<br>- filter (string)<br>- options (string) | Returns number of objects returned by QueryObjects. "sortby" and "attributes" cannot be defined as they don't affect the result. |
| QueryObjectsFirstAttribute (string) | - query (string)<br>- filter (string)<br>- sortby (string)<br>- attributes (string)<br>- options (string) | Attribute value of the first object of queried objects. Only one attribute is defined in "attributes". |
| QueryObjectsSum (double) | - query (string)<br>- filter (string)<br>- attributes (string)<br>- options (string) | Sum of queried objects of numerical attributes. Only one attribute is defined in "attributes". Error is thrown if attribute values are not numerical. |
| QueryObjectsUnique (string) | - query (string)<br>- filter (string)<br>- sortby (string)<br>- attributes (string) | Concatenates unique attribute values of all queried objects using defined separator. Only one attribute is defined in "attributes". |

|  |  |  |
|---|---|---|
|  | - options (string)<br>- separator (string) |  |
| SubAttributesAsArray (array) | - objectid (string)<br>- attribute name (string)<br>- subattribute name (string)<br>- filter expression (string) | Returns attribute values (as an array) which are presented in more complex structures. This type of attributes are e.g. *graphicalproperties*, *customattributetypes* and *properties*. The filter expression is for selecting desired rows. In the filter expression there are following arguments:<br>- all sub tag attributes (with their tag names)<br>- argument **ordernumber** having values 0, 1, … for xml tag order number.<br>Example, following graphicalproperties attribute:<br>&lt;graphicalproperties&gt;<br>&lt;symbol id="920256947" x="790" y="190" width="400" height="30" instanceid="2029031131" /&gt;<br>&lt;symbol id="2009602777" x="230" y="640" **width="200"** height="40" **instanceid="0"** /&gt;<br>&lt;symbol id="368985118" x="730" y="40" width="100" height="60" instanceid="369716419" /&gt;<br>&lt;/graphicalproperties&gt;<br>Calling:<br>**SubAttributesAsArray([instanceid], 'graphicalproperties', 'width', '[instanceid]=0')**<br>returns 200. |
| SubAttributesAsDataset (dataset) | - objectid (string)<br>- attribute name (string)<br>- columns (array) | Returns attributes which are presented in more complex structures as a dataset. This type of attributes are e.g. graphicalproperties, customattributetypes, properties and hotspots. Dataset columns (names of the sub attributes) are listed as an array (in lower case). |

# 9 Dataset functions

Dataset is a table like object containing multiple named columns and multiple rows. Dataset can contains zero rows, but at least one column must exist. Dataset may contain any type of data. Dataset is like a table in relational database or a worksheet in Excel.

Usually dataset functions don’t change the input dataset, but create new dataset (exceptions are mentioned). Thus the input datasets may also be used in other functions.

Dataset functions get their idea from SQL language.

| Function | Parameters | Description |
|---|---|---|
| AddColumn (dataset) | - dataset<br>- columnName (string)<br>- expression (string)<br>- compare mode (boolean) | Adds a new column to the dataset. All other column values are available as arguments in the expression. Also there is an argument **rowordernumber**, which is the row order number starting from 0.<br>In the compare mode (true), there are also values of the previous and next rows available as arguments with suffixes **_previous**, **_current** and **_next**. For example if there are columns **column1** and **column2**, and |

| | | |
|---|---|---|
| | | adding a new column **column3**, there are arguments column1_previous, column2_previous, column3_previous , column1_ current, column2_current, column1_next and column2_next. |
| AddDatasetRow (dataset) | - column 1 value (object)<br>- column 2 value (object)<br>- … | Adds a new row to a dataset with the provided values. A row is added to the provided dataset, i.e. no new dataset is created. |
| BuildHierarchy (dataset) | - dataset<br>- instance id column name (string)<br>- object id column name (string)<br>- parent object id column name (string)<br>- Hierarchy id column name (string)<br>- Hierarchy parent id column name (string)<br>- level column name (string)<br>- filtering formula (string)<br>- sorting formula (string) | Builds an element hierarchy based on data containing element instances. The data is provided in a dataset, where there are following columns: instance id, object id and parent object id. The object id can be derived from instance id.<br>The function constructs a new dataset where lines are in the hierarchy order. The function can handle situations where also other than the bottom level objects have been instantiated; it means that the result hierarchy will contain same instances multiple times. Following new columns are added:<br>- hierarchy id (explained below)<br>- hierarchy parent id (referring to the hierarchy id)<br>- level (top level is 0, the level directly below top is 1, …)<br>The **hierarchy id** means an id that is unique for the whole hierarchy. To enable this following technique is used: If there is an instance id PG.x.y.z which appears multiple times, the first hierarchy id is PG.x.y.z, the second is PG.x.y.z.1, the third is PG.x.y.z.2, and so on. |
| CreateDataset (dataset) | - column names (string array) | Creates a new dataset with provided column names. The created dataset contains no rows. |
| DatasetCell (object) | - dataset<br>- column name (string)<br>- row number (int) | Gets an item from a dataset from the named column and from the named index (first row is number 1). Returned type is the type of the cell. |
| DatasetSize (int) | - dataset | Number of rows in a dataset. If dataset is null, -1 is returned. |
| Distinct (dataset) | - dataset | Removes duplicate rows, i.e. rows that contains identical data. |
| Except (dataset) | - dataset1<br>- dataset2 | Result dataset contains rows that are in the first dataset but not in the second dataset. The datasets must have same columns. |
| From (dataset) | - query (string)<br>- filter (string)<br>- sortby (string)<br>- attributes (string)<br>- options (string) | Returns QPR Web Service’s QueryObjects result as a dataset. Columns get their names from the result data. Column names can be changed with QueryObjects syntax **attribute(as="ColumnName")**. |

| | | |
|---|---|---|
| FullOuterJoin (dataset) | - left dataset<br>- right dataset<br>- matching expression (string) | Full outer join of two datasets. The expression have all column names as arguments. The datasets cannot have columns with same names. |
| GroupBy (dataset) | - dataset<br>- list of grouped columns (string array)<br>- list of combined column names (string array)<br>- list of combine expressions (string array) | Groups the dataset. Parameters:<br>- 1. parameter is the dataset to group<br>- 2. parameter is an array of columns to group<br>- 3. parameter is an array of combined column names<br>- 4. parameter is combine expression (e.g. count, sum, average or number of rows). All expressions have as arguments an array of objects to combine. Parameter names equal to column names.<br>Example: GroupBy([dataset1], ItemsToArray('modelname', 'typename'), ItemsToArray('count', 'objectnames'), ItemsToArray('ArraySize([name])', 'Concatenate([name])')) |
| InnerJoin (dataset) | - left dataset<br>- right dataset<br>- matching expression (string) | Inner join of two datasets. The expression have columns names as available arguments. The datasets cannot have columns with same names. |
| Intersect (dataset) | - dataset1<br>- dataset2 | Intersect of two datasets. Result dataset contains only those rows that are in both datasets. The datasets must have same columns. |
| LeftJoin (dataset) | - left dataset<br>- right dataset<br>- matching expression (string) | Left join of two datasets. Expression have columns names as available arguments. The dataset cannot have columns with same names. |
| Matrix (dataset) | - dataset<br>- row expression (string)<br>- column expression (string)<br>- value expression (string)<br>- grouping expression (string)<br>- row column name (string) | Builds a matrix based on the provided dataset. Row and column expressions are executed for each row of the dataset, and the data is positioned to rows and columns in the matrix based on the row and column expression values.<br>The value for the input dataset row is determined by the value expression.<br>The grouping expression has an input argument "value", containing an array of all calculated value expressions for the matrix cell.<br>*Row column name* defines the column containing matrix row names.<br>Columns are sorted as an alphabetical order (except the matrix row name is the first). |
| MetricsMeasureValues (dataset) | - element id's (string array)<br>- series symbols (string array)<br>- value column names (string array)<br>- value attribute name (string) | Returns Metrics measure values as a dataset from multiple periods (dataset rows) and multiple series or measures (dataset columns). Each period id will have a separate row, i.e. data from different measures are combines in a same row if the measures have a same period level.<br>Element id's, series symbols and value column names in the result dataset are provided as a separate arrays. All arrays size must be the same |

| | | |
|---|---|---|
| | | The result dataset will have **columns**, **periodid**, **startdate**, **enddate** and **periodname**.<br>Valid value attribute names are **value** and **prettyvalue** (these are from QPR Web Service’s *values* attribute). Data in result dataset are strings, and in case "value" attribute is used, they can be converted to numeric. |
| RemoveColumns | - dataset<br>- columnNames (string array) | Removes the specified columns from a dataset. Example:<br>RemoveColumns([dataset1], ItemsToArray('name')) |
| RightJoin (dataset) | - left dataset<br>- right dataset<br>- matching expression (string) | Right join of two datasets. Expression have columns names as available arguments. The dataset cannot have columns with same names. |
| SortBy (dataset) | - dataset<br>- sorting definition (string) | Sorts the dataset. Sorting is defined as comma separating the sorted columns. Optionally **asc** (default) or **desc** can be added after the column name. E.g. **attribute1 asc,attribute2 desc** |
| Split (dataset) | - dataset<br>- split column name (string)<br>- split expression (string) | Splits every row of a dataset into multiple rows based on the split expression. The split expression must return an array containing splitted items for a single row. Thus, the number of items in the array determines, to how many rows a single row is splitted. The splitted items are stored in a new column.<br>The split expression has all dataset column values as arguments.<br>Note that the splitting may also result to a single row or even zero rows, if the split expression returns an array of one or zero items.<br>Example:<br>Split([dataset], 'splitted', 'StringToArray([column_1], \',\')') |
| Transpose (dataset) | - dataset<br>- column name for headers (string) | Transposes a dataset, i.e. changes its rows to columns and columns to rows.<br>If column name for headers is not provided, in the transposed dataset first column name is **headers** and it contains the header names of the original dataset. Rest of the header names are **column_1**, **columns_2**, …<br>If column name for headers is provided, data in that column is used as headers for the new dataset (instead of generated headers described above). |
| Union (dataset) | - dataset1<br>- dataset2 | Union of two datasets. Results dataset contains all rows of the input datasets. The datasets must have same columns. |
| Where (dataset) | - dataset<br>- expression (string) | Filters out all rows in the dataset where expression is evaluated as false. Expression has all columns as arguments. |

# 10 XML functions

| Function | Parameters | Description |
|---|---|---|
| SelectXMLAttribute (string) | - XDocument or XElement<br>- XPath expression (string)<br>- namespace prefix (string)<br>- namespace URI (string)<br>- attribute name (string) | Gets an XML attribute value from an XML element that is selected from an XML document using an XPath expression. An empty string is returned if no element matches or if the attribute is not found. |
| SelectXMLAttributes (string array) | - XDocument or XElement<br>- XPath expression (string)<br>- namespace prefix (string)<br>- namespace URI (string)<br>- attribute name (string) | Gets a list of XML attribute values from a list of XML elements that are selected from an XML document using an XPath expression. An empty set is returned if no element matches. |
| SelectXMLElement (XElement) | - XDocument or XElement<br>- XPath expression (string)<br>- namespace prefix (string)<br>- namespace URI (string) | Selects an element from an XML document using XPath expression. The namespace definition is for the XPath expression. Null value is returned if no element matches. |
| SelectXMLElements (XElement array) | - XDocument or XElement<br>- XPath expression (string)<br>- namespace prefix (string)<br>- namespace URI (string) | Selects an array of elements from an XML document using XPath expression. The namespace definition is for the XPath expression. An empty set is returned if no element matches. |
| XmlAttribute (string) | - XElement<br>- attribute name (string)<br>- default value (string) | Gets an XML attribute value from an XML element. Empty value (or optional default value) is returned if the attribute is not found. |
| XmlDocument (XDocument) | - XML data (string)<br>- XML schema (string)<br>- custom error message (string) | Constructs an XML document from a string and validates it. Type of the returned object is **XDocument**. The XML data is provided as the first parameter and an XML schema as a second parameter. If the XML document is not compatible with the schema (i.e. valid), an error is thrown. |
| XPathEncode (string) | - string | Encodes a string to be suitable for an XPath expression. The function adds quotes ("") if needed. |

# 11 Binary data functions

File data functions are for getting file contents as a byte array from different sources. There are also functions for getting media types of files.

| Function | Parameters | Description |
|---|---|---|
| HttpFileData (byte array) | - url (string)<br>- alternateUrl (string) | Fetches a file as byte array through an http(s) using the provided url address. The address must point directly to the file resource. If no resource is found from url, an alternateUrl is tried. |

| | | |
|---|---|---|
| HttpFileMediaType (string) | - url (string)<br>- alternateUrl (string) | Media type of the file fetched using HttpFileData function with a same parameter. If no resource is found from url, an alternateUrl is tried. |
| LoadFileFromDisk (byte array) | - location (string) | Loads a file from the local disk drive as byte array. To take into account data security, avoid implementation where users are able to select the file to load. Users need to have read access to the file. |
| QprEmbeddedFileData (byte array) | - object (string)<br>- attribute (string)<br>- options (string) | Gets an embedded file from QPR system as byte array using QPR Web Service's **GetBinaryData** operation with provided parameters. More information http://kb.qpr.com/qpr2015-1/index.html?getbinarydata.htm. |
| QprGraphData (byte array) | - object (string)<br>- options (string) | Gets an image file as byte array using QPR Web Service's **GetGraph** operation with provided parameters. |
| QprGraphInfo (string array) | - object (string)<br>- options (string) | Get image width and heigth as an integer array. |
| QprGraphMediaType (string) | - object (string)<br>- options(string) | Media type of the file fetched using qprEmbeddedFileData function with same parameters. |

## 12 Other functions

| Function | Parameters | Description |
|---|---|---|
| ApplicationSetting (string) | - setting name (string) | Returns application setting value by its name. Settings depend on the application where the expression language is used. E.g. application settings for applications, that are part of QPR Web Services Extensions, are defined in web.config of QPR Web Services Extensions. |
| ApplicationVersion (string) | [none] | Returns a version number of the application using the expression language, e.g. DWR or DWV. |
| Coalesce (object) | - object<br>- object, … | Returns first of the objects (counting from left to right) which value is<br>- for strings, first string that is not null or empty string, or<br>- for other than strings, first value that is not null.<br>Minimum of two parameters are needed.<br>If all parameters are nulls or empties, the first parameter is returned. |
| DataType (string) | - object | Return the type of provided parameter, e.g. int, string, datetime. |
| DiagramPath (string array) | - object id (string)<br>- attribute name (string)<br>- parent relation (string) | Returns an array of EA/PD diagram paths for an object or an object instance. An array is returned as there can be several diagram paths if the object has several instances or if diagrams have been instantiated. Value of the provided attribute is used |

| | | |
|---|---|---|
| | - separator (string)<br>- options (string) | to identify an object in the diagram path (usually object "name" is used).<br>Parent relation is the relation attribute name for getting parent objects (usually “parentobjects” is used). Also a "separator" character must be provided (usually "/"). "Options" is for QueryObjects operation for getting parent objects. |
| DynamicExcelReport (byte array) | - report name (string)<br>- parameter names (string array)<br>- parameter values (object array) | Runs a Dynamic Excel Reports report. |
| DynamicWebViews (byte array) | - template name (string)<br>- parameter names (string array)<br>- parameter values (object array) | Runs a Dynamic Web Views template. |
| DynamicWordReport (byte array) | - report name (string)<br>- parameter names (string array)<br>- parameter values (object array) | Runs a Dynamic Word Reports report. |
| Eval (object) | - expression (string) | Evaluates an expression. |
| ExecuteRecursion | - return expression (string)<br>- recursion expression (string)<br>- recursion initial value (object)<br>- exclude traversed (boolean) | Executes a recursion based on provided expressions. There are two expressions:<br>- **return expression**: Determines the value a recursion step returns. The expression contains an argument **recursionresult** which is an array containing the return expressions of all the one level below recursion steps.<br>- **recursion expression**: Determines the next level recursions item. Must return an array. The value of this array item will be given to the next step of the recursion. The expression contains an argument **currentrecursionstep** which is the recursion value of the current step.<br>The **recursion initial value** is the value where the recursion starts.<br>Exclude traversed means that items that have already been encountered during the recursion are excluded from next level recursion items.<br>Example:<br>**ExecuteRecursion('1 + Sum([value])', 'QueryObjects(\'[\' + [value] + \'].childobjects\', \'\', \'\', \'id\', \'\')', 'PG.123.456')** |

| | | |
|---|---|---|
| ExecuteSearch (string array) | - search configuration (XDocument) | Executes a search for QPR objects. The search is based on a search configuration provided as an xml document. The schema of the XML document is described in Appendix 1. |
| ExpressionLanguageVersion (string) | [none] | Returns expression language version number, i.e. version of QPR Suite Accelerator .Net Tools. QPR Suite Accelerator .Net Tools can be upgraded by replacing the QPRSuiteAcceleratorTools.dll in the QPR Web Services Extensions folder under IIS with a new one. |
| GenerateNumberArray | - start (int)<br>- increment (int)<br>- count (int) | Generates an integer array starting from **start** integer, using defined **increment** till the **count** is generated. |
| GetLog (string array) | - dateformat (string) | Gets WSE log. |
| If | - condition (boolean)<br>- true value (object)<br>- false value (object) | Usual programming conditional statement. The **condition** is evaluated, and if the condition is true, the **true value** is returned; otwerwise it returns the **false value**. If the condition is evaluated to a null value, the false value is returned. The false value is optional, and in that case null is returned if condition is false.<br>If the condition is true the false value (expression) is not evaluated and vice versa.<br>Example: **if([variable1] = 2, 'value is two', 'value is something else')** |
| IsNull (boolean) | - object | Return true if the value is null, otherwise false. |
| IsNumeric (boolean) | - string | Returns true if the input string can be converted to a numerical value, otherwise false. |
| Loop (object) | - array<br>- expression (string) | Loops through an array and calculates an expression for every iteration. The function gives a value of the last iteration’s expression as a result. If array length is zero, a null value is returned. Following arguments are available in the expression:<br>- **value**: Item in the array.<br>- **previousresult:** Result of previous iteration’s calculated expression. For the first iteration, this value is null.<br>- **index**: Iteration order number starting from 0.<br>Example: **Loop(StringToArray('4,2,3', ','), 'coalesce([previousresult], 0) + [value]')** (gives 4+2+3=9) |
| NullValue | - [none] | Return null value. |
| OdbcReadData (dataset) | - connection string<br>- query (string)<br>- connection timeout (int) | Reads data from an ODBC source and returns the data in a dataset. To use the function, **connection string** and **query** need to be defined. Connection strings can be found in https://www.connectionstrings.com |

| | | |
|---|---|---|
| | - query timeout (int) | Connection and query timeouts are optional; they can be changes if needed.<br>Appropriate ODBC drivers needs to be installed.<br>Example for Excel file (read all from sheet "Sheet1"): **OdbcReadData('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)}; DBQ=C:\\test.xlsx;', 'Select * from [Sheet1$]')** |
| RaiseError | - error message (string) | Raises (throws) an error and shows an error message that is passed as a parameter. This can be used if there is a more complex logic for parameter validation. |
| ReportsMenu | - menu configuration(s) (string or string array)<br>- template name (string) | Return html for report items.<br>Template must have parameter reportitems, which is a dataset containing report definitions.<br>visibility, reporturl, reportname, visiblemessage, disabledmessage, target, width, height, scroll, preventcaching, x, y |
| SwitchCase (object) | - control expression (object)<br>- condition1 (object)<br>- value1 (object)<br>- condition2 (object)<br>- value2 (object)<br>- default value (object) | Conventional programming "switch" statement. If **control expression** equals **condition1**, **value1** is returned and so on. Control expression may be string, integer, double or date. If no condition matches, the **default value** is returned. Note that the number of parameters must be an equal number (4, 6, 8, …).<br>Example: **SwitchCase([variable1], 1, 'value is one', 2, 'value is two', 'value is something else')** |
| UsageLog (string) | - Log line (string)<br>- Log file (string)<br>- Log mode (string) | Writes a line of text to a local file in disk. The function can be used e.g. for usage logging. Parameters:<br>- **Log line** is a line of text to append to the log file.<br>- **Log file** is the full path and name for the log file.<br>- **Log mode** is optional can be one of the following:<br>○ **1**: Errors are skipped.<br>○ **2**: Error message is returned by function as a string.<br>○ **3**: An exception is thrown by the function. This means that the expression calculation fails.<br>If no errors occur, the function returns an empty string. When using Windows authentication, all users need to have write access to the log file. The function may be used in an Expression or Variable tag in Word reports. Note the possible security and performance issues when writing files. |

| | | |
|---|---|---|
| | | Example: **UsageLog(CurrentDateTime () + ';' + CurrentUser() + ';' + ReportName(), 'C:/Logs/QPRReportsLog.txt')** |
| WsSessionId (string) | [none] | Returns QPR Web Service session id, if there is a valid session. If not, returns an empty string. |

# 13 Dynamic Word Reports functions

These functions are only available when used in Dynamic Word Reports.

| **Function** | **Parameters** | **Description** |
|---|---|---|
| ReportTemplateData (byte array) | templatepath | Gets a DWR template Word file as byte array based on provided template path. |
| ReportName (string) | [note] | Returns the name of the current report. |

# 14 Dynamic Web Views functions

These functions are only available when used in Dynamic Web Views templates.

| **Function** | **Parameters** | **Description** |
|---|---|---|
| ExistsTemplate (boolean) | - template name (string) | Returns true if the template exists; otherwise false. |
| ReadTemplate (string) | - template name (string) | Returns template contents as a string. Template must be referenced using absolute path (see DWV documentation).<br>It's possible to read any type of files by adding the file suffix to the file name (for **tpl** files, no suffix is added) Example: to read file schema.xsd in the root folder, use path **/schema.xsd**. |
| TemplateName | [none] | The current template, i.e. the name of the template where the expression is run. |
| TemplateParameters | [none] | Returns all template parameters as a dictionary (string, object). |
| TemplatePath | [none] | The current template's path, i.e. the folder path of the template where the expression is run. The folder path starts from the templates root folder. No starting or ending slashes are part of the path. |

# 15 Appendix 1: ExecuteSearch function configuration

The search is based in a defined **scope**, which is a set of objects where the search is targeted. A scope can be e.g. all published models, all certain types of elements. The scope also includes element attributes where the search is targeted. A scope consist of multiple **scope parts**, which are QPR Web Services queries.

The objects defined by the scope are filtered using a **criteria** (the search results are the matching objects). Criteria can be any expression containing and, or, not and parenthesis.

| **XML element** | **Description** | **Attributes** |
|---|---|---|

| | | |
|---|---|---|
| executesearch (1) | parent: none | **scopecombining**: and, or |
| scopepart (1…n) | Defines a scope part, which contains a web service query, transformation and attributes.<br>Possible parent element: executesearch | **query**: QPR Web Services query<br>**options**: QPR Web Services query options<br>**transformation**: transformation relation name |
| criteria (1…n) | Defines criteria for search. There can be criteria elements inside other criteria elements to form a hierarchical structure representing expression calculation order.<br>Criteria can be defined under executesearch for a criteria for all criteria parts, or the criteria can be defined under a certain criteria part.<br>Possible parent elements: executesearch, scopepart, criteria | **type:** one of the following:<br>- and (one to many sub criteria)<br>- or (one to many sub criteria)<br>- not (one sub criterion)<br>- text (no sub criteria)<br>**searchtext:** searched string. Used when type=text.<br>**attribute:** searched QPR Web Service attribute. Used when type=text.<br>**comparisontype:** one of the following: contain, begin, is. Used when type=text<br>**matchcase:** true or false |
| attribute (0..n) | Defines searched attributes related to a scope part. Parent element is scopepart | **name:**  name of the attribute |

Example:

```
<?xml version=”1.0” encoding=”utf-8”?>
<executesearch xmlns=”http://www.qpr.com/QPRSuite/ExecuteSearch” scopecombining=”and”>
  <scopepart query=”[PG.1221241820].Ominaisuus” transformation=”” options=”QueryModelingLanguage=EN”>
    <criteria type=”and”>
      <criteria type=”text” searchtext=”haku12” attribute=”name” comparisontype=”begin” matchcase=”true”/>
      <criteria type=”text” searchtext=”haku3” attribute=”sanastontermi.value” comparisontype=”is” matchcase=”false”/>
    </criteria>
  </scopepart>
  <scopepart query=”[PG.1221241820].Käsite” transformation=”käsitteenominaisuus”>
    <criteria type=”and”>
      <criteria type=”text” searchtext=”haku1” attribute=”name” comparisontype=”contain” matchcase=”false”/>
    </criteria>
  </scopepart>
</executesearch>
```

Schema:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.qpr.com/QPRSuite/ExecuteSearch"
xmlns:search="http://www.qpr.com/QPRSuite/ExecuteSearch" elementFormDefault="qualified">

  <xs:complexType name="criteria">
    <xs:sequence>
      <xs:element name="criteria" type="search:criteria" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="type" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="text"/>
          <xs:enumeration value="and"/>
          <xs:enumeration value="or"/>
          <xs:enumeration value="not"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="searchtext" type="xs:string"/>
    <xs:attribute name="attribute" type="xs:string"/>
    <xs:attribute name="comparisontype">
      <xs:simpleType>
        <xs:restriction base="xs:string">
```

```
          <xs:enumeration value="contain"/>
          <xs:enumeration value="is"/>
          <xs:enumeration value="begin"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="matchcase" type="xs:boolean"/>
  </xs:complexType>

  <xs:element name="executesearch">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="scopepart" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="criteria" type="search:criteria" minOccurs="0" maxOccurs="1"/>
              <xs:element name="attribute" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:attribute name="name" type="xs:string"/>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:attribute name="query" type="xs:string" use="required"/>
            <xs:attribute name="options" type="xs:string"/>
            <xs:attribute name="transformation" type="xs:string"/>
          </xs:complexType>
        </xs:element>
        <xs:element name="criteria" type="search:criteria" minOccurs="0" maxOccurs="1"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```