@Formula-Language

Short Reference

by udo junghans (junghans<at>eras.de)

version 0.9 (2.05.2013)

1. Syntax

A := 2; einfache Zuweisung Abschluss immer mit ";"

Zwischen Werten, Strings, Keywords, Variablen immer ein Leerzeichen

FIELD CityUpper := "…"; Zuweisung zu einem Feld "CityUpper" (Feld wird im Dokument angelegt, falls noch nicht vorhanden)
@SetField(field;content) Synonym zu FIELD
@GetField("FieldName"); Feldinhalt auslesen
DEFAULT FieldName := ""; schreibt einen Defaultwert

2. (Logische) Operatoren

:=	Zuweisung (Assignment)
[]	Listenindex
:	Trennzeichen zwischen Listenelementen
+	Positiv
-	Negativ
(*)*	(permutierte) Multiplikation
(*)/	(permutierte) Division
+	Addition, Zusammenfügen von Strings
(*)+	(permutierte) Addition
(*)-	(permutierte) Subtraktion
(*)=	(permutiertes) gleich
<>	ungleich
!=	ungleich
=!	ungleich
><	ungleich
(*)<>	(permutiertes) ungleich
(*)<	(permutiertes) kleiner als
(*)>	(permutiertes) größer als
(*)<=	(permutiertes) kleiner oder gleich
(*)>=	(permutiertes) größer oder gleich

permutierte operationen:
(A : B) + (C : D) = (A+C : B+D)
(A : B) *+ (C : D) = (A+C : A+D : B+C :B+D)

!	logisches NOT
&	logisches AND
|	logisches OR

Prioritäten der Abarbeitung: () > ^ > */ > +-
Oder von links nach rechts

Mustersuche in Strings:
@LIKE (*string* ; *pattern*)
stringExpr Like *patternString*
Muster:

?	irgendein Zeichen
#	irgendeine Ziffer 0 … 9
*	irgendein String 0 – viele Zeichen
[*characters*]	Irgendein Zeichen oder Zeichenbereich
[!*characters*]	Irgendein Zeichen oder Zeichenbereich die hier nicht angegeben sind

@Contains(*string* ; *substring*) ist substring in string enthalten

3. Kommentare

REM "Hier könnte Ihr Kommentar stehen.....";
REM ["remarks"];
REM {....};
OutOfOrder := {};

4. Benutzerkommunikation

@Prompt
@DialogBox
@PickList
@StatusBar(Message$)

5. Flusskontrolle

@IF(Bedingung1 ; WENN_TRUE1 ; ELSE_Aktion);
@IF(Bedingung1 ; WENN_TRUE1 ; Bedingung2 ; WENN_TRUE2 ; ELSE_Aktion); bis zu 99 Bedingungen möglich
@Do(anweisung1; anweisung2; ...anweisung250); Block von Anweisungen anstatt einer einzelnen Anweisung in @IF !! Letzte Anweisung innerhalb DO-Block darf nicht mit „;" abgeschlossen werden.
@For(*initialize* ; *condition* ; *increment* ; *statement1* ; *..;.statement252*);
@Return(); Stoppt die weitere Formelabarbeitung, liefert Wert in Klammern zurück.
@IF(True;@Return("");"");
@DoWhile(*statement1;... statement254*; *Bedingung*); Bedingung wird **nach** Ausführung geprüft
@While(*condition* ; *statement1* ; *...statement254*); Bedingung wird **vor** Ausführung geprüft

6. Converting data types

Variablen müssen den korrekten Typ haben, um Operationen korrekt ausführen zu können.

@Char(number) Konvertiert eine IBM® Code Page 850 Codenummer in das entspr.
@IsNumber(value) gibt 1 zurück, wenn value eine Zahl ist, sonst 0
@IsText(value) 1 wenn Text sonst 0
@IsTime(value) 1 wenn Zeitangabe sonst 0
@Text(value[; format]) Konvertiert alles zu Text, ein format kann angegeben werden
@TextToNumber(string) Konvertiert Text in Zahl
@TextToTime(string) Konvertiert Text in Datums-/Zeitangabe
@TimeToTextInZone(date-time ; time zone) wie TextToTime mit Angabe der Zeitzone
@TimeZoneToText(time zone) Zeitzonenangabe in einen String.
@ToNumber(string or value) Konvertiert alles in eine Zahl
@ToTime(string or date-time value) Konvertiert eine Zahl oder Zeitangabe in eine Zeitangabe.

7. Math functions

@Abs(number) Betrag
@ACos(cosine) inverser Cosinus
@ASin(sine) inverser Sinus.
@ATan(tangent) inverser Tangens.
@ATan2(x; y) zweiparametrischer inv. Tangens
@Cos(angle) Cosinus in RAD
@Exp(number) Potenzierung zur Basis e
@FloatEq(number ; number ; range) Vergleich zwei Zahlen innerhalb eines Vertrauensbereiches

@Integer(num or list) Entfernt Nachkommastellen
@Log(number) Logarithmus zur Basis 10.
@Ln(number) Natürlicher Lorarithmus (Basis e).
@Max(number or list; number or list;...) Größter Wert in Liste
@Min(number or list; number or list;...) Kleinster Wert in Liste
@Modulo(number or list ; number or list) Berechnet den Divisionsrest.
@Pi Kreiszahl Pi
@Power(base ; exp) Potenzierung (Wurzel).
@Random Zufallszahl zwischen 0 und 1.
@Round(number or list [; factor]) rundet auf Integer, mit factor kann Genauigkeit angegeben warden (100 rundet auf 100; 0.01 rundet auf 2 Stellen hinter Komma.
@Sign (number) gibt Vorzeichen der Zahl zurück; 1 für positive, 0 für negative
@Sin(angle) Sinus in RAD
@Sqrt (number) Quadratwurzel
@Sum(num or list; num or list; ...) Summe der Zahlen
@Tan(angle) Tangens in RAD

8. List functions

@Compare(*textlist1; textlist2; [options])* Paarweiser Vergleich zweier Textlisten
@Count(list) Anzahl der Elemente in Liste, liefert 1 wenn Nullstring oder keine Liste.
@Elements(list) Anzahl der Elemente in Liste, liefert 0 wenn Nullstring oder keine Liste.
@Explode(string[; separators ; includeEmpties]) Konvertiert einen String in eine Liste, Leerzeichen, Kommas, Semikolons oder NeuZeilen geben die Elementgranzen an. Mit *separators* kann Trenner explizit angegeben werden. Mit includeEmpties = @TRUE werden aufeinanderfolgende Trenner zu leeren Elementen konvertiert.
@Explode(dateRange) Konvertiert eine Zeitraumangebe in einzelne Zeitpunkte.
@Implode(list[; separator]) Konvertiert eine Liste in Text, *separator* gibt den Trenner für die einzelnen Elemente des Ergebnisses an, default ist: „ „
@IsMember(string oder list; list2) Untersucht, ob string oder list in list2 vorkommt, Liefert True oder False.
@IsNotMember(string oder list; list2) Umkehrung von @IsMember
@Keywords(list1; list2) Locates words in list1 that match words in list2. Word separators are " , ? ! ; : [ ] { } < >
@Keywords(list1; list2; separator) As above, but the second parameter specifies the word separators.
@Max (number or numberlist) Returns the largest number in the list.
@Max (number or numberlist ; number or numberlist) Returns the larger number of two numbers or a number list of the largest numbers resulting from a pair-wise computation of two number lists.
@Member(value ; list) Determines the position of a value in a string list.
@Min (number or numberlist) Returns the smallest number in the list.
@Min (number or numberlist ; number or numberlist) Returns the smaller number of two numbers or a number list of the smallest numbers resulting from a pair-wise computation of two number lists.
@Nothing Adds nothing to a transformed list.
@Replace(list1; list2; list3) Replaces values in list1 that match values in list2 with the corresponding values in list3.
@Sort(list ; [order]) Sorts a list. Order is [Ascending] (default) or [Descending].
@Subset(list ; n) Extracts n number of values from the list. Use -n to extract right to left.
@Transform(list ; name ; formula) Applies a formula to each element of a list.
@Unique(list) Removes duplicate values from a string list.
@Unique Returns a random, unique text value.

9. String functions

Pattern matching → look (logical Operators)
@UpperCase(string) / **@LowerCase** Converts a string to UPPERCASE / lowercase
@ProperCase Converts the first of each word to a capital letter
@Trim Removes leading and trailing spaces from a text string, removes duplicate spaces between words.
@Text Converts a number, date, or time to a text.
@LeftBack(str;int|str) / **@RightBack(str;int|str)** / **MiddleBack(str;substr/int;substr/int)** / **@Left(str;int/substr)** / **@Right(str;int/substr)** / **@Middle(str;substr/int;substr/int)** Searches a string and returns int characters from left/ right/ middle (side) of string or until substr / between substrings.
@Begins(str;substr**)** / **@Contains(**str;substr**)**/ **@Ends(**str;substr**)** Determines whether a substring is stored at the beginning / middle / end of another string
@Replace(*sourcelist* **;** *fromlist* **;** *tolist* **)** Performs a find-and-replace operation on a text list.
@ReplaceSubstring(*sourceList* ; *fromList* ; *toList*) replaces text in sourceList, source, fron and to can be lists of strings
@Word(*string*;*separator*;*number*) cut out nth word from string(list) separated by separator, negative n takes nth word from right

10. Regular expressions

@Matches(*string* **;** *pattern* **)**
Pattern: Text or text list. The pattern you want to scan for in *string* surrounded by quotation marks. May contain wildcard characters and symbols (see table below). The following symbols require a preceding backslash unless the pattern is enclosed in braces { } as a set: ?, *, &, !, |, \, +. The symbols require two preceding backslashes instead of one if the pattern is specified as a literal. This is because the backslash is an escape character in string literals, so "\?" passes "?" to the matching engine, where it is treated as a wildcard, while "\\?" passes "\?" to the matching engine, where it is treated as a question mark character.

Symbols:
C matches C or c
? Matches any single character
* Matches any string (any number of characters)

{ABC} Matches any character in set ABC
{A-FL-R} any character in the sets A...F and L...R
+C Matches any number of occurrences of C (or c)
! Complements logical meaning of the pattern (logical NOT)
| Performs logical OR of two patterns
& Performs logical AND of two patterns

11. Datums-Funktionen

@Date([y; m; d; h; m; s]) Liefert Jahr, Monat, Tag, Stunde, Minute und/oder Sekunde
@Day (date) Liefert den Tag des angeg. Datums
@Hour(time-Date) Stunde
@Minute(time-date) Minute
@Month(date) Liefert den Monat des angeg. Datums
@Now Liefert das aktuelle Datum und Uhrzeit
@Now([ServerTime]; serverNames) Datum des Servers auf dem die aktuelle DB liegt oder Datum eines anderen LotusServers
@Second(time-date)
@Time([time-date] [y; m; d; h; m; s]) liefert Zeitangabe zum Input
@Today Liefert das aktuelle Datum
@Tomorrow Liefert das morgige Datum
@Weekday(date) Liefert den Wochentag des Datums (Sonntag=1, Montag=2...)
@Year(date) Liefert das Jahr des angeg. Datums
@Yesterday Liefert das gestrige Datum
@Zone[(timeDate)] liefert die Zeitzoneneinstellung des derzeitigen Rechners oder eines angegebenen Datums

@Accessed Liefert das Datum des letzten Zugriffs auf das Dokument
@Created Datum der Erstellung des Dokuments
@Modified Datum und Zeit der Letzten Änderung/Speicherung des Dokuments

@Adjust(time-date; y; m; d; h; m; s) Berechnet ein neues Datum durch Angabe von Zeitschritten
@BusinessDays Liefert die Anzahl der Arbeitstage in einem Zeitraum

12. Notes.ini-settings / Systemeinstellungen

Set Settings:
@Environment("My";"0");
ENVIRONMENT My := “0”;
@SetEnvironment("My"; "0"); writes $My=0

Get settings:
My := @Environment("My"); to get content of $My

Abfrage von Registry-Einträgen (nur Win)
@RegQueryValue("HKEY_LOCAL_MACHINE"; "SOFTWARE\\...\\"; "szDatVersion")

13. Infos über Arbeitsumgebung

@LocationGetInfo (
[HomeServer]
[CatalogServer]
[SametimeServer]
[NamePreference]
[MailProtocol]
[WebRetriever]
[BookmarksFileName]
[InternetMailAddress]
[AreaCode]
[UNID]
[Country]
[InternationalPrefix]
[LongDistancePrefix]
[LongDistancePrefix])
@DbBuildVersion