Programowanie IBM Domino aplikacji

Tworzenie formuł
cz. I

dr inż. Jacek Grekow
Wydział informatyki, Politechnika Białostocka

Co to są formuły?

- Prosty język do programowania w IBM Domino Designer

Sklada się z:
- Elementów języka formuł
- @functions i @Commands

Przykłady

- @Created
- FirstName + “” + LastName
- @Command ([FileCloseWindow])

Przykłady

Gdzie wykorzystujemy formuły?

The Formula language can be used in
- agents
- replication formulas
- views
- forms
- actions
- Pages
- fields
Gdzie wykorzystujemy formuły?

Formula Window

Gdzie wykorzystujemy formuły?

Script area - the error message is at the bottom of the Script area.

Gdzie wykorzystujemy formuły?

properties box

Gdzie wykorzystujemy formuły?

Script area

Formuły składają się z:

- Zmiennych
- Stałych
- Operatorów
- Słów kluczowych
- @Funkcji

Zmienne

- FirstName
- FirstName + . + Lastname
- @ProperCase (FirstName)
Stałe

- Tekstowe np. „Kowalski”
- Numeryczne
- Czasowe np.: [01/18/2017 09:45:30]

Operatory

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Suma</td>
</tr>
<tr>
<td>-</td>
<td>Różnica</td>
</tr>
<tr>
<td>*</td>
<td>Cenę * 3</td>
</tr>
<tr>
<td>/</td>
<td>Cenę / 2</td>
</tr>
<tr>
<td>+</td>
<td>Cena + Dodatek</td>
</tr>
<tr>
<td>-</td>
<td>Cena – Roznica</td>
</tr>
<tr>
<td>=</td>
<td>Równość</td>
</tr>
<tr>
<td>!=</td>
<td>Nie równość</td>
</tr>
<tr>
<td>&lt;</td>
<td>Mniejsze</td>
</tr>
<tr>
<td>&gt;</td>
<td>更大的</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Mniej lub Równa</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Duże lub Równa</td>
</tr>
<tr>
<td>(</td>
<td>()</td>
</tr>
</tbody>
</table>

Logiczne

- Status = „zatwierdzony”
- KlienciDocelowi = „Studenci”

Numericzne

- Status = „zatwierdzony”
- KlienciDocelowi = „Studenci”

Kodowanie

- @Left (Phone; 3)
- @If(@IsNewDoc; “New Chapter”; “Chapter “ + cChNumber)

Formula Language Keywords

- **REM**
  - **FIELD**
  - **SELECT**

**FIELD**

- Assigns a value to a field, creating the field if it doesn't exist.
- Sets any text following as nonexecuting documentation. The text must be in quotes or curly braces.
- The syntax is:
  - `FIELD fieldname := value;`
  - if the field exists, its value is overwritten by `value`.

**SELECT**

- The syntax for `SELECT`:
  - `SELECT argument` is used to select documents based on the conditions supplied in the argument.
  - `@All` is a frequently used argument and is the default for a new views and agents. If the result of the argument returns a true value for the document, it will be selected.

**REM**

- Comments with REM
  - REM is used to comment your formula.
  - Its syntax is pretty simple: REM "text" or REM {text}
  - REM [Include the text "New Chapter" or "Chapter" plus the chapter number ];
  - @If(@IsNewDoc; "New Chapter"; "Chapter “ + cChNumber)
This view selection formula displays documents created with the NH form in which the offer is not withdrawn and applies a fairly complex series of date restrictions based on the day of the week and the shift.

@Funkcje
- Istnieje ponad 200 funkcji
  - Używa się do :
    - Formatowania tekstu
    - Generowania i formatowania czasu i dat
    - Obliczania numerycznych wartości
    - Obliczania wartości w listach
- Zwracane wartości:
  tekstowe, numeryczne, czas, logiczne

Typy @Funkcji
- Arithmetic: @Abs, @Integer, @Round, @Modulo
- Client information: @BrowserInfo, @ClientType
- Data retrieval: @DbColumn, @DbLookup, @DbCommand
- Database information: @DbTitle, @DbName, @DbManager
- Date and time: @Date, @TextToTime, @Time, @Hour, @Year, @Day, @BusinessDays
- Document information: @DocLength, @Attachments, @DocumentUniqueID, @SetDocField(documentUNID; fieldName; newValue)
- Document status: @IsDocBeingSaved, @IsNewDoc, @IsDocBeingEdited, @DocLock
- Field values: @GetDocField, @GetField, @GetProfileField, @GetValue, @ThisName, @ThisValue
- Iterative: @DoWhile, @For, @While List @Count, @Member, @Elements, @Subset, @Contains, @Trim,
  @implode, @explode, @sort, @DbColumn, @DbLookup
- Logical: @True, @False, @Success, @IsTime, @IsNumber Mail @MailSend, @MailDbName, @MailSavePreference
- String: @Right, @LeftBack, @MiddleBack
- User information: @UserName, @UserRoles, @NameLookup, @UserAccess
- User input: @Prompt, @DialogBox, @PickList

Zasady pisania formuł
- Separatory
  Dwukropek – oddziela składniki listy
  “sport”: “szachy”: “muzyka”
- Średnik
  Odziela argumenty funkcji
  @if( condition ; action ; else-action)
- Nie rozróżnia małych i dużych liter
  Za wyjątkiem słów kluczowych REM, SELECT

Typy @Funkcji cd.
- Field values: @GetDocField, @SetField, @GetProfileField, @GetField, @ThisName, @ThisValue
- Iterative: @DoWhile, @For, @While List @Count, @Member, @Elements, @Subset, @Contains, @Trim, @implode, @explode, @sort, @DbColumn, @DbLookup
- Logical: @True, @False, @Success, @IsTime, @IsNumber Mail @MailSend, @MailDbName, @MailSavePreference
- String: @Right, @LeftBack, @MiddleBack
- User information: @UserName, @UserRoles, @NameLookup, @UserAccess
- User input: @Prompt, @DialogBox, @PickList

Soup Language Keywords
SELECT cd
REM {Normally, notify 2 days in advance. On Friday, notify 3 days in advance.}
REM {On Thursday, notify third shift 4 days in advance}
REM {Restrict “Offer Withdrawn” from the view}
SELECT (Form = “NH” & cWithdrawn != “Offer Withdrawn”) &
{If(@Weekday(@Today) = 6; dStart <= @Adjust(@Today; 0; 0; 3; 0; 0; 0); dStart <= @Adjust(@Today; 0; 0; 2; 0; 0; 0)}
{If(@Weekday(@Today) > 5 & cShift = “Third” ; dStart <= @Adjust(@Today; 0; 0; 4; 0; 0; 0); dStart <= @Adjust(@Today; 0; 0; 2; 0; 0; 0))
 & dStart >= @Adjust(@Today; 0; 0; -2; 0; 0; 0))

This view selection formula displays documents created with the NH form in which the offer is not withdrawn and applies a fairly complex series of date restrictions based on the day of the week and the shift.
@Funkcje do operacji na tekstach

• Locating Substrings Within Strings
  @Begins() Determines whether a string begins with another string  
  @Contains() Determines whether a string contains another string  
  @Ends() Determines whether a string ends with another string

• Extracting Substrings From Strings
  @Left() Returns leftmost characters of a string, searching from left to right  
  @LeftBack() Returns leftmost characters of a string, searching from right to left  
  @Middle() Returns characters from the middle of a string, searching from left to right  
  @MiddleBack() Returns characters from the middle of a string, searching from right to left  
  @Right() Returns rightmost characters of a string, searching from left to right  
  @RightBack() Returns rightmost characters of a string, searching from right to left

• Comparing Strings
  @Like() Compares two strings (similar to @Match() but is ANSI SQL–compliant)  
  @Matches() Compares two strings

• Manipulating Strings
  @Length() Returns the length of a string  
  @LowerCase() Converts a string to lowercase  
  @ProperCase() Converts a string to proper case, capitalizing the first letters of words  
  @UpperCase() Converts a string to uppercase  
  @Repeat() Repeats a string  
  @ReplaceSubString() Replaces elements of a string  
  @Text() Converts other data types to text strings  
  @Trim() Removes leading and trailing blanks

@Funkcje logiczne

• Many of these @Functions are related to the state of a document, such as @IsDocBeingEdited or @IsNewDoc. For example, placing @IsNewDoc in the Hide When of an action button will hide it when a document is first created.
  @If(@IsNewDoc, "New Service Request", "Old Service Request")

• For example, you might want to sort and categorize a view on a date field that exists in some, but not all, documents. If the field containing the date doesn’t exist, you want to sort it on the date of the document’s creation. The following formula tests whether the date field exists and uses it if it does; otherwise, it uses @Created.
  jdDate := @If(@IsAvailable(dCall) & dCall != "", dCall, @Created);

@Funkcje logiczne

• The @Functions @IsText, @IsNumber, and @IsTime test the data type of a field. @IsMember and @IsNotMember are used with lists to determine whether a value or list exists within another list.

• For example, if you want to alert the user that a value entered in a field is incorrect or that a field with a required entry was left blank, you can use @Failure in a field’s Input Validation event, as in the following example:
  @If(cSubject = "", @Failure("You must enter a subject in order to save this document!"); @Success)

Lists logical @Functions

@IsAgentEnabled Returns True if the agent is enabled.
@IsAppInstalled Tests whether Admin or Design client is installed.
@IsAvailable Tests whether a field exists in a document.
@IsCategory Returns True if any item in a row of a view is a category.
@IsDocBeingEdited Returns True if the document is in edit mode.
@IsDocBeingLoaded Returns True if the document is being loaded.
@IsDocBeingMailed Returns True if the document is being mailed.
@IsDocBeingRecalculated Returns True if the document is being refreshed (recalculated).
@IsDocBeingSaved Returns True if the document is being saved.
@IsDocTruncated Returns True if the document is truncated.
@IsError Returns True if a value has an error condition.
@IsExpandable Returns True if a row on a view is expandable.
@IsMember Returns True if a text item or text list is a member of another text list.
@IsModalHelp Returns True if the document is a modal help document.
@IsNewDoc Returns True if the document has just been created.
Lists logical @Functions

- `Yes` Returns `True`, or 1.
- `No` Returns `False`, or 0.
- `Businesses` Returns `True`, or 1.
- `IsNotMember` Returns `True` if a text item or text list is not a member of another text list.
- `IsNumber` Returns `True` if the value is numeric.
- `IsTime` Returns `True` if the value is text.
- `IsResponseDoc` Returns `True` if the document is a response document.
- `IsText` Returns `True` if the value is text.
- `IsTime` Returns `True` if the value is a time-date value or a time-date list.
- `IsUnavailable` Returns `True` if the field does not exist in the document.
- `IsValid` Returns `True` if all validation formulas are successful.
- `IsNull` Returns `True` if the value is a single text value that contains nothing; otherwise, returns `False`.
- `IsVirtualizedDirectory` If virtualized directories are enabled on the current server, returns `True`. Otherwise, returns `False`, or 0.
- `true` Returns `True`, or 1.
- `false` Returns `False`, or 0.

@Funkcje Data-time

The following function converts the number of the month to the text equivalent:

@Select(@Month(dSales); “January”; “February”; “March”; “April”; “May”; “June”; “July”; “August”; “September”; “October”; “November”; “December”)

@Funkcje Data-time

- The function `@Now` returns the current time-date of the machine on which the formula is executed.
- `@SetField(“cDocHistory”; cDocHistory : “Notification emailed to ” + jcSendTo + “ on ” + @Text(@Now; “S2”))`

@Funkcje Data-time

- Converting Time-Date Values:
The `@Text()` function is very versatile because of the level of control you can exert over the conversion process. The syntax is the following:

  `@Text(time-date; parameters)`

- Working with Date Math:
  All date math is expressed in seconds, so `@Tomorrow - @Yesterday` returns 172,800.
  If you want to know the number of days between two time-date values, simply divide the result by the magic number: 86,400.

@Funkcje Data-time

- "Quarter" = @Text(@Integer(@Month(dSales)-1) / 3))

@Funkcje Data-time

- QAccessed, @Created, and @Modified - @Functions properties of documents.
- @Now, @Today, @Tomorrow, and @Yesterday all return specific time-date values, as indicated by their names.
- You use the other @Functions to manipulate time and date values in formulas.
- The function `@Adjust()` is particularly useful because it enables you to adjust a time-date value in the past or into the future.
- `@Adjust(DateTime; Year; Month; Day; Hour; Minute; Second; [InLocalTime] | [InGMT])`
- To adjust an invoice due date to 30 days in the future, you can use `@Adjust(dDue; 0; 0; 30; 0; 0; 0)`
- To adjust it for one month in the future, you can use the following formula:
  `@Adjust(@Now; 0; 0; 0; 0; 0; 0; 0)`
- @Functions can be particularly useful in views in which you want to sort documents by the date.
Time-Date @Functions

Text() In this context, converts time-date values to text
TextToTime() Converts a text representation of a time-date value to a time-date value
Time() Returns the time component of a time-date value
Today() Returns today’s date
Tomorrow() Returns tomorrow’s date
Weekday() Returns the weekday number of a time-date value
Year() Returns the year of a time-date value
Yesterday() Returns yesterday’s date
Zone() Returns the time zone component of a time-date value

Using Hide When Formulas

if you want to display a check box stating Resigned on the same row as some other information and have it display when the actual value is Resigned, check Hide Paragraph If Formula Is True and include a formula such as the following:
Resigned != "Resigned"

Another typical use for hide formulas is to hide actions based on roles assigned to users. If only administrators should access a button, create a role called Admin and use a formula such as the following:
Contains(@UserRoles; "Admin")
Of course, you can combine this with other values. For example, you can create an action button that enables an application administrator to change the status of a document. You might not want anyone outside of the Admin role to use this button, and you also might not want the button displayed when the document is new. The following formula accomplishes this:
IsNewDoc | Contains(@UserRoles; "Admin")

Writing View Formulas

- The default view selection is SELECT @All
- SELECT (Form = "Building Owners") | (Form = "BO") & ($Contains(BuildingsOwned; "Building"))

Writing Column Formulas

- Some typical types of column formulas are the following:
  - Counting documents in categories
  - Counting documents in categories that meet certain criteria
  - Summing field values and subtotaling by category
  - Converting field aliases into a text equivalent
  - Sorting and categorizing by date parts, such as year, month, and quarter
  - Displaying an icon instead of a field value

Converting Field Aliases

Document Status Sorting

```c
if(cDocStatus = "New"; 1;
   cDocStatus = "Pending Approval"; 2;
   cDocStatus = "Approved"; 3;
   cDocStatus = "Completed"; 4;
   cDocStatus = "Denied"; 5;
   99)
```

Editable Field Formulas

- Input translation formula
- Input validation formula

Help

Formula Language @Functions A-Z:

Trendy Language A Functions A-Z
The document shows the names and usage of all the functions, in alphabetical order. It also includes examples, where appropriate. For more information, see the documentation and tutorials.

When does the function cost (per-day)?
When does the function cost (per-day)?
For more information, see the documentation and tutorials.

Language Feedback on help?