WINWORD$

WINWORD$ opens and creates an RTF document, saves it as Word document and returns the name of the document

Syntax

WINWORD$ Template.rtf(Target$, Mode%=0, 1, 2, [InpVar])

Arguments

- Template.rtf is the name of the RTF template file created by MS Word. The following does apply:
  - It is necessary to include the ".rtf" extension;
  - The file should be stored on the current Applications directory;
  - The RTF document should be checked by Quaestor on any impropper RTF code. Do this through the menu item *Tools>RTF Template Check...*
- Target$ is a string expression containing the name of the document to be created (use for instance the TIME() and ORCA() functions for the generation of this name).
- Mode% are options:
  - Mode% = 0: do NOT start MS Word to show the resulting document;
  - Mode% = 1: start MS Word with the resulting document;
  - Mode% = 2: do NOT start MS Word and leave document as .rtf;
- InpVar is an optional list of parameters used in Template.rtf. While creating the expression, the parser will search the template for any parameters and include these in the expression automatically. Any parameters that are not automatically traced by the expression parser should be included because the parameters in the template will not be determined on evaluation of the expression unless they are in the expression itself.

Remarks

1. Although the document can only be saved as .doc if MS Word is installed. MS Word is not necessary to carry out some of the basic template functionality as this is done on a Rich Text Document format. However, when Word automation is required (in case of creating tables using the @RUNWORDMACRO() function, or building a document based on several separate documents or Word macro's), MS Word will be required.
2. To use all the advantages of the MS Word integration, the use of VBA code should be allowed (can be found in the security settings).
3. On some systems unsolvable MS Word errors may occur resulting into an unfortunate Quaestor error. When this is the case, please select mode%=2 and open the document separately.
4. In multi-case solutions, MS Word is not started per case, even if Mode%=1.
5. Please note that WINWORD$ only works with RTF documents and not with .DOC, .TXT, etc.
6. Target$ should not contain any path information, the output file is ALWAYS written to the current Report directory. If path information is included in Target$, it is removed. Preferably, do not use the name of your template as Target$.
7. The WINWORD$ function returns the file name being the value of Target$. In the event of multi case document generation, "_" + CaseNr% is added automatically, e.g. DataFax_4.doc for the 4th case unless you have controled this multi case dependency yourself by adding "_"+ORCA (1) to your expression.
8. The simplest way to create a template file is by means of MS Word and save the document as rtf.
9. On any location where you want to insert values computed by Quaestor in the template, it should be indicated using format statements between "#". See the TEMPLATE$() function for options.

Qnowledge has provided a macro with the Quaestor distribution to assist with the creation of the RTF templates in MS Word. The file is by default installed in the "Qnowledge\Quaestor" directory and called "QnowledgeForm.dot".

To use this macro, place the file in the directory "startup" of Microsoft Office. Usually this is directory is in: "C:\Program Files\Microsoft Office\Office.. \STARTUP". Please note that you might also need to change your Word macro security settings because the macro as no Thrust certificate.
After placing the file in the indicated directory an additional button is provided. For further explanation on the macro please click here.

1. To use document parts into a master document, you have to generate the document parts and add tags to these document parts using their file name as parameter (so including "#") in the master document (or any other sub-level). The sub-documents should exist before the master document is generated. This is done easiest by adding parameters for generating the document tags and for generating the sub-documents in the expression for generating the master document. Note that this functionality uses Word automation.
2. Please note that, comparable to creating tables in Word with @RUNWORDMACRO() function by means of a VBA macro based on a TeLiTab data set, you can execute any VBA macro inside the document on a specified place. The only thing you have to do is to refer in the document template to a parameter which contains a reference to the macro file (*.bas). A powerful aspect is the possibility to create the macro file itself as part of the process in the knowledge base. In this way, you are able to use very strong Word automation through standard VBA functionality and the TEMPLATE$() function. However, please be aware that Quaestor will not have knowledge of the VBA macro's itself, the Knowlege Engineer will stay responsible for the proper operation of the macro's as part of the document generation.
3. Since version 2.49.2 you can also execute your macro's after document generation as part of the generation itself.
   Add @RUNWORDMACRO to the dataslot of the relation with the parameter name containing the reference to a macro file or reference to the macro file itselt.

The general process of creating the total functionality

The following steps are required to generate a document with Quaestor:

Step 1)

Create an RTF document and store it on the current applications directory (either of the knowledge base, located in Kbs_<KnowledgeBaseName>\Applic, or the general Applic directory in My Knowledge).

For extensive information on document formatting options, please revert to the TEMPLATE$() function.

Step 2)

Verify the template document by using the menu option *"Tools>RTF Template Check" in Quaestor*.

This is necessary to remove parasite RTF codes in the format statements (just follow the instructions). Please note that MS Word includes RTF codes as it were on arbitrary locations which might destroy some of the code you have created to include functionality and parameters.

After you select the file (it shows the .rtf files on the current application directory), the check is performed and any changes are reported in a .log file. Quaestor reports the name of the original file, creates a .log file and makes a back-up of the original with a time stamp (2.46.2 and higher).

Step 3)

Load the knowledge base in which to include the document generator (or start with an empty KB (Newqkb))

Select in the knowledge browser the option New Relation and type expression.

Close the expression editor by selecting Save. If you select the new relation in the Browser, in the frame viewer you will see all existing parameters in the template are included in the expression. You may observe that template parameters in expression are not included. This is because the presence of parameters in logical expressions do not need to exist in the current solution to be used in the expression (this parameter being PENDING might also be information to be used). So if you want certain parameters to be present before the document is created, make sure these are included in the expression.

After this, your document generator is ready for use.

Step 4)

To generate a document, double click on the parameter connected to the expression to start the dialogue. Quaestor will either ask for values of the above parameters or calculate them (if possible). If the WINWORD$ function is invoked and finished, with the Mode% being 1, MS Word will open with the resulting document. This document is a normal Word document as any one created within Word itself. Note that the resulting document starts as a RTF document which will be saved as .DOC in Word, resulting in a considerable reduction of file size. Reports created by WINWORD$ can be of enormous size if, for instance graphical information is included. The original created document remains on the current Report directory.

After completion of the solution, the parameter connected to the expression will contain the name of the output document as returned/contained by Target$, without path information. All (intermediate) results of Quaestor sessions including program output and any generated documents are stored on the current (project's) Report directory.

Examples

A. The template

The template document can contain any graphical and/or markup information deemed necessary. As mentioned above, on any location where you want to insert values computed byQuaestor in the template, it should be indicated using format statements between "#". See the TEMPLATE$() function for options.

In the example of a template:

TO FAX : #FAXNO$#
COMPANY: #COMPANY#
ATTN : #CLIENTCONTACT$#
SUBJECT: #SUBJECT$#

Dear ..
The power consumption at the design speed is #VDES|F6.2# knots.
#NS=2[1]#The vessel has two propellers[1]
With best regards,
#PROJMANNAME$#

In this example NS=2 is an expression that controls everything between the [1] labels. When true, the text between "[1]#" and "propellers[1]" is added, if false the part between "#NS" and "propellers[1]" is removed.

Please check the rtf document with Quaestor. After checking the above template, Quaestor will show the resulting log made of the required changes Quaestor made. This might be as followed:

#FAXNO$#
#FAXNO$#
#COMPANY#
#CLIENTCONTACT$#
#SUBJECT$#
#VDES|F6.2#
#NS=2[1]#
[1]
#PROJMANNAM}{\f2\lang1033\cgrid0 E}{\f2\lang1033\cgrid0 $#
RTF code in parameter or constraint, removal attempted =>
RTF code in parameter or constraint, removal attempted =>
#PROJMANNAME$#

The example shows that there was some RTF code that had to be removed.

B. The expression

Using the above template, we make an expression using the template with the WINWORD$ function:

```
DataReport$ = WINWORD$ DataFax.rtf(FaxFile$,
"1")
```

After saving the expression you will see that the additional parameters are added, resulting into the following expression:

DataReport$ = WINWORD$ DataFax.rtf(FaxFile$, 1, FAXNO$, COMPANY, CLIENTCONTACT$, SUBJECT$, VDES, PROJMANNAME$)

You will see that NS has not been included. If you want NS to be present, provide the first relation as:

DataReport$ = WINWORD$ DataFax.rtf(FaxFile$, 1, NS)

which will give

DataReport$ = WINWORD$ DataFax.rtf(FaxFile$, 1, NS, FAXNO$, COMPANY, CLIENTCONTACT$, SUBJECT$, VDES, PROJMANNAME$)

C. Advanced options

Document parts

When you want to add a separate document part as part of the template add a document tag to this document part. Using the example, you can do this in this way:

TO FAX : #FAXNO$#
COMPANY: #COMPANY#
ATTN : #CLIENTCONTACT$#
SUBJECT: #SUBJECT$#

Dear ..
The power consumption at the design speed is #VDES|F6.2# knots.
#NS=2[1]#The vessel has two propellers[1]
#DocumentTag$#
With best regards,
#PROJMANNAME$#

In the template we added #DocumentTag$#. This parameter will contain the document tag required to include the separate document with the correct name. By setting this all up parametrically it will stay working for one or more cases. So we will add the following relation:

DocumentTag$="#"+SubDocument$+"#"

In which SubDocument$ is the parameter generating the sub document, comparable to DataReport$ above.

Furthermore, add DocumentTag$ and SubDocument$ to DataReport$.

Addressing through objects or TeLiTabs

When you use several objects, data from these objects can be addressed in the template. This done as follows:

#ObjectName.ParameterName#

or

#ObjectName.ObjectName.ParameterName#

etc.

You will receive the data for the case of ParameterName which is selected at the moment of document generation. For a specific case you have to add a number, like for case 3:

#ObjectName.ObjectName.ParameterName.3#

When you want to receive all case values for ParameterName you have to write:

#ObjectName.ObjectName.ParameterName.0#

You will receive all cases separated with a CrLf.

When you want to address values in TeLiTab data, it is slightly different. You can address this in the template as follows:

#TeLiTabName.ParameterName#

If the object or TeLiTab contains several cases for the parameter, you will receive all parameters as a list of values with carriage return linefeeds (CrLf) in between. When you want a nice comma separated sentence instead, you have to create a separate parameter for the list of values and use additional attributes in the data slot of the expression.

For example, within a "Calculation" object you have done calculations on one or more Rigs with a "RigNumber". Place the summary parameter "RigNumber$" in the template:

#RigNumber$#

And create a relation:

RigNumber$=Calculation.RigNumber

with in the dataslot of this expression:

@TEXTDELIMITER:", "
@TEXTLASTDELIMITER:" and "

The attributes indicates that comma's should be used between values and that the final two values should have " and " between them.

Add tables

This is very much like adding sub-documents. Use the @RUNWORDMACRO() function to create a Word macro for a table based on TeLiTab input (when required create the TeLiTab first with the TELITAB#() function). In the template add the reference to the parameter that creates the table macro and also add this parameter to the relation that uses the template.

Quick links: Functions overview | Attribute overview | Constants overview | Dimensions overview