













## Knowledge Advisor

-  [Preface](#)
-  [What's New?](#)
-  [Getting Started](#)
-  [Basic Tasks](#)
-  [Advanced Tasks](#)
-  [Workbench](#)
-  [Description](#)
-  [Quick Reference](#)
-  [Case Study](#)
-  [Glossary](#)

 P1

 P2



# Preface

A major issue that affects every industry in which products are developed is the retention and reuse of knowledge. While every organization produces data and information about its products, transforming that data into an accessible knowledge base is difficult. The tools used throughout the product development process typically do not help retain the knowledge of how and why decisions are made, nor do they provide convenient access to that knowledge. Knowledge is the embodiment of experience and data in a directly useable form. How you get home from your office is knowledge. A map showing the way is data that has to be interpreted. Through experience, you know to use different routes shown on the map depending on various conditions such as the weather or the time of day. You have used the map and other information to create knowledge. Likewise, in product development, understanding how a product is intended to function and how to apply that understanding to create an appropriate product is knowledge. The models and drawings of the product are part of the data that defines the product, but they don't convey how or why someone created the product in a specific way. It is difficult to embed knowledge in a design. It takes extra effort to write out or codify the decision process so that it can be passed as knowledge from one person to another. The tools that are most often used in product development also need to be capable of capturing knowledge, and they must allow product developers to apply that knowledge to their product designs.

CATIA Version 5, in addition to embodying a significant new architecture that has been evolving over a number of years, also provides some very interesting capabilities that allow designers to drive their design processes from associative specifications and to integrate specifications into their product development processes. It gives product developers ways to create specifications and rules, save them, and reuse them.



[Highlights](#)



[Knowledgeware Documents](#)



[About Samples](#)

# Highlights

The CATIA knowledgeware capabilities are based on special features that you add to a document to enrich its data with knowledge-based information. The features you are going to create and manipulate in a CATIA knowledgeware application are divided into three categories:

- the parameters
- the relations
- the behavior.

Parameters and relations help you design a document with respect to a particular know-how. Once these features have been created, the Knowledge Inspector can be run on a document to determine impacts and dependencies.

The behavior is a particular feature. It is written in Visual Basic language and executed only when a feature replies to a given event.

Here are some highlights about the features you are going to manage with the Knowledge Advisor product.

**Parameters** Parameters define the properties of a document. When used within relations, they act like arguments. Parameters are given a name, a type and a value. In place of a value, a parameter can be defined by a relation. The parameter is then constrained, you won't be able to modify directly its value.

Parameters are divided into two categories:

- The intrinsic parameters: they are part of the document and depend on the application. The purpose of a knowledgeware application is not to create or delete these parameters but to specify how these parameters can be constrained by relations.
- The user parameters: they are irrespective of the document basic definition. They should be considered as extra pieces of information. Knowledgeware applications quite often use these parameters to add knowledge to a document and to define relations.

User parameter capabilities are provided to all CATIA users. You can create and manage user parameters with the CATIA Infrastructure product.

## **Relations**

Knowledgeware applications manage four types of relations:

- *the formulas*  
a formula defines how a parameter is to be calculated with respect to other parameters. Formula capabilities are provided to all CATIA users.
- *the design tables*  
data contained in tables referred to as design tables can be used to manage document parameters. Design table capabilities are provided to all CATIA users.
- *the rules*  
a rule is a set of instructions which conditionally executes a group of statements depending on a context or depending on the values of some parameters.
- *the checks*  
a check is a set of instructions whereby you are warned that some statements are fulfilled or not. Checks have no impact on parameter values.

Rule and check capabilities are only provided to those of you who have the Knowledge Advisor product installed.

## **Knowledge Inspector**

The Knowledge Inspector is a tool to analyze impacts and dependencies when a number of parameters and relations have been added to a document. It offers two options:

- What If  
What is going to be modified if a given parameter is changed
- How To  
What parameters should be modified in order to modify a given parameter.

The Knowledge Inspector requires the Knowledge Advisor product.

## **Behavior**

A behavior is a set of operations associated with a feature. It is executed as a reply to an event. A behavior is defined by:

- the feature it is applied to
- a set of operations
- the event which triggers the list of operations.

A behavior is written in Visual Basic language.



# The CATIA Knowledge Advisor Documentation

This documentation covers the knowledgeware capabilities provided both in the Infrastructure and the Knowledge Advisor products. It is divided into the following parts:

- [Getting Started](#)

Introduces the CATIA knowledgeware capabilities by working through a simple example. Little detail is given in this part. Its purpose is to give beginners a global idea of the concepts.

- [Basic Tasks](#)

Provides the bulk of information needed by users building knowledgeware applications. Is mainly based on interactive tasks such as creating relations. Syntax rules to be used when programming relations are also detailed.

- [Advanced Tasks](#)

Focuses on design table aspects and behavior. Describes programming functions which allow you to access parameter values in design tables. Explains how to write a behavior.

- [Workbench Description](#)

Describes the Knowledge Advisor icons.

- [Case Study](#)

Applies the CATIA Knowledge Advisor capabilities to a deep groove ball bearing.

- [Quick Reference](#)

Summarizes the interactions to perform when manipulating knowledgeware features.

- [Glossary](#)

Gives some useful definitions.



Highlights

Knowledgeware Documents



About Samples



# About Samples

The material provided in this documentation is mainly based on step-by-step instructions. Generally, the exercises are easy to complete because of the basic instructions provided. The exercise in [Case Study](#) is a little more complicated and requires a good knowledge of the Part Design and knowledgware capabilities, especially if you want to redo it from scratch.

To do the exercises which illustrate the knowledgware principles, you should have installed a certain number of samples. These samples are listed below. Refer to the "Infrastructure" documentation for information on how to access these samples.

## Starting or Resulting Documents

KwrStartDocument.CATPart	Infrastructure	"Getting Started" & Basic tasks.
KwrForRuCh.CATPart	KnowledgeAdvisor	Resulting document of "Getting Started".
KwrParam0.CATPart	Infrastructure	Basic tasks.
KwrMaterial0.CATPart	Infrastructure	Basic tasks.
KwrImport.CATPart	Infrastructure	Basic tasks.
KwrFormula0.CATPart	Infrastructure	Basic tasks.
KwrFormula1.CATPart	Infrastructure	Basic tasks.
KwrFormula2.CATPart	Infrastructure	Basic tasks.
KwrProgramDT.CATPart	Infrastructure	Illustrates how to program the design table access.
KwrDesignTable0.CATPart	Infrastructure	Advanced tasks.
KwrDesignTable1.CATPart	Infrastructure	Advanced tasks.
KwrDesignTable2.CATPart	Infrastructure	Advanced tasks.
KwrBallBearing0.CATPart	KnowledgeAdvisor	Case study.
KwrBallBearingResult.CATPart	KnowledgeAdvisor	Case study.

## Tabulated Text Files

TxCompanyFile0.txt	Infrastructure	Basic tasks.
KwrBallBearingImport.txt	KnowledgeAdvisor	Case study.

## CATIA Macro

KwrMacro1.CATScript	KnowledgeAdvisor	Case study.
---------------------	------------------	-------------

## Microsoft® Excel Tables

KwrCreatedDesignTable.xls	Infrastructure	Advanced tasks.
KwrBallBearing.xls	KnowledgeAdvisor	Case study.
ExCompanyFile0.xls	Infrastructure	Basic tasks.

# What's New?

## Knowledgeware - Infrastructure

### Functions to Access a Design Table

Two new functions: [CloserInfConfig](#), [CloserSupConfig](#)

### Tools->Options->Knowledge

[Copy/Pasting parameters](#) - [Parameters written in non-latin characters.](#)

## Knowledge Advisor

### Programming Rules

[Temporary variables](#)

### Using the Knowledge Inspector

Enhanced: [What If Mode](#), [How To](#)

### Creating a Behavior

New task: [Creating a Behavior](#)

# Getting Started: CATIA Knowledgeware at a Glance

We introduce the CATIA Knowledgeware capabilities by working through a simple example. Since many of the knowledgeware concepts are unfamiliar to you at this point, little detail is given. You just have to follow step-by-step the instructions described below to get a first appreciation of the CATIA knowledgeware principles.

Before starting, check the following options:

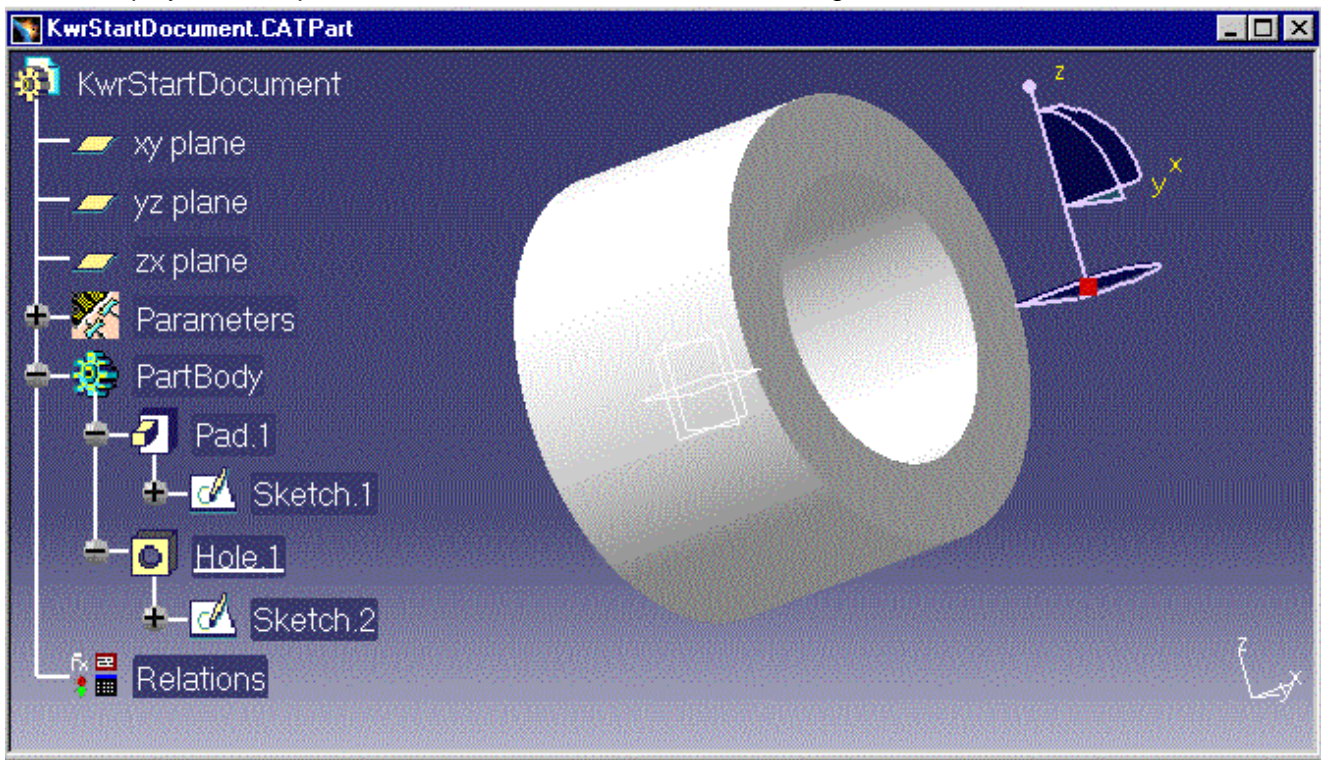
- Tools->Options->Part->Display->Parameters
- Tools->Options->Part->Display->Relations
- Tools->Options->General->Knowledge->Parameter Tree View->With Formula.

## Parameters

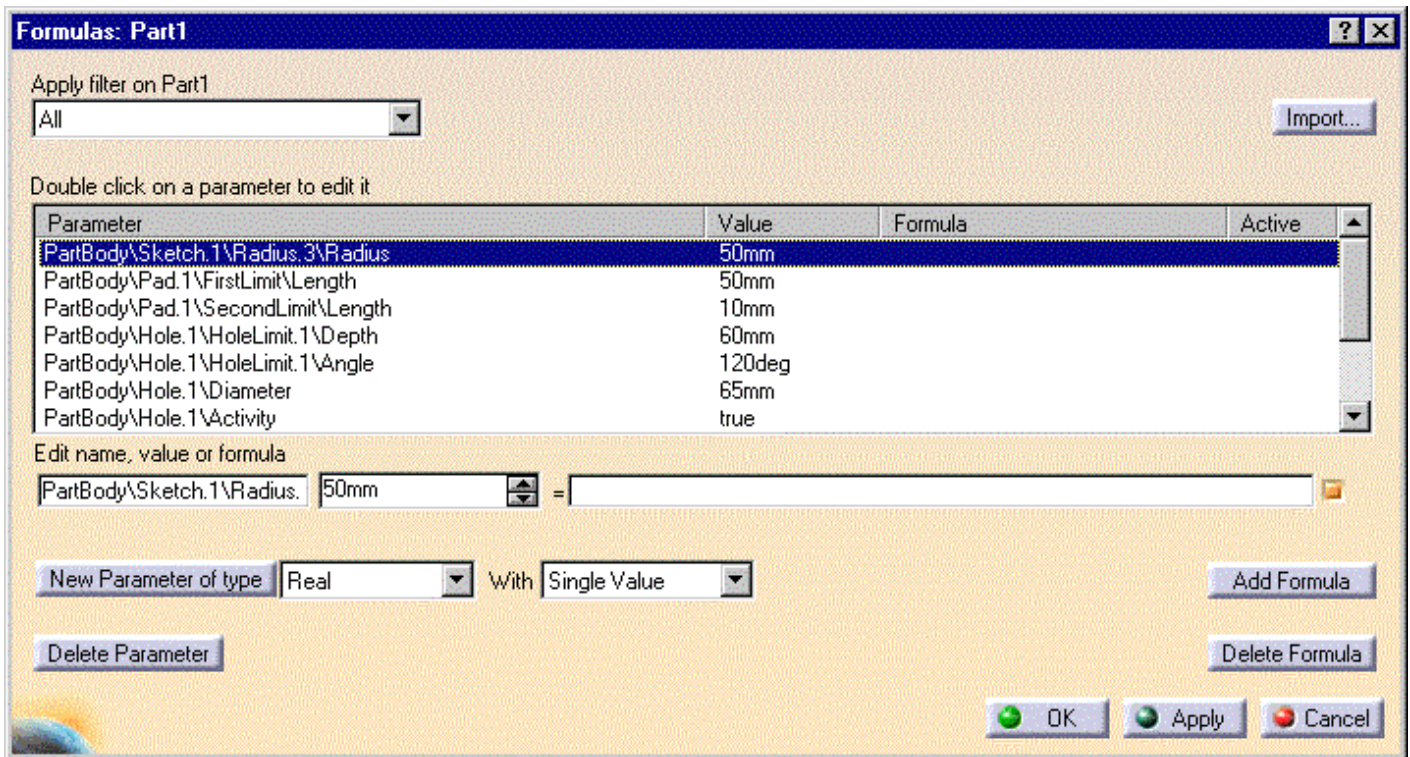
This task can be performed by all CATIA users. Its purpose is to give a brief outline of what parameters are and how you can manipulate them. For a fuller outline of the parameter-related tasks, see [Parameters](#).



1. Open the KwrStartDocument.CATPart document. This document is a hollow cylinder. The document parameter list can be displayed. New parameters can also be added to the existing ones.



2. Click the  icon. The "Formulas" dialog box is displayed.




3. In the 'New Parameter of type' list, select the Length item, then click 'New Parameter of type'.
4. In 'Edit name, value or formula', replace the Length.1 string with PadLength. Click Apply. A new parameter is added to the document parameter list both in the "Formulas" dialog box and in the specification tree. You have just created a *user parameter*.
5. Click OK in the "Formulas" dialog box to terminate the dialog. **Keep your document open and proceed to the next task.**

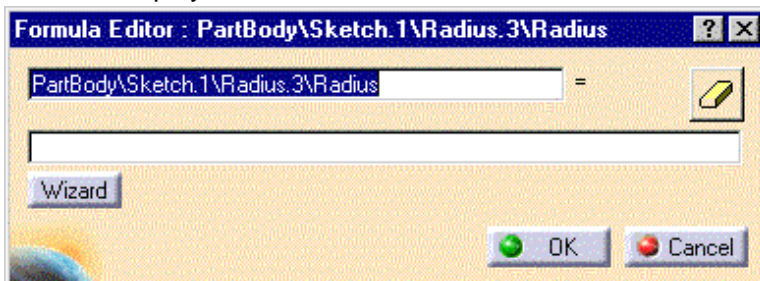


## Formulas

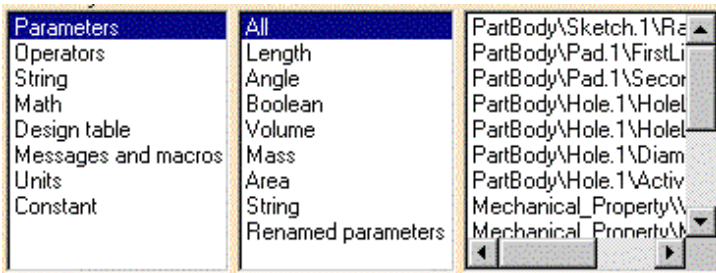
This task can be performed by all CATIA users. Its purpose is to explain how a parameter can be constrained by a formula. Go to [Formulas](#) for more information on formula-related tasks.



1. Click the  icon. The "Formulas" dialog box is displayed.
2. In the parameter list, select the PartBody\Sketch.1\Radius\Radius item, then click 'Add Formula'. The "Formula editor" is displayed.

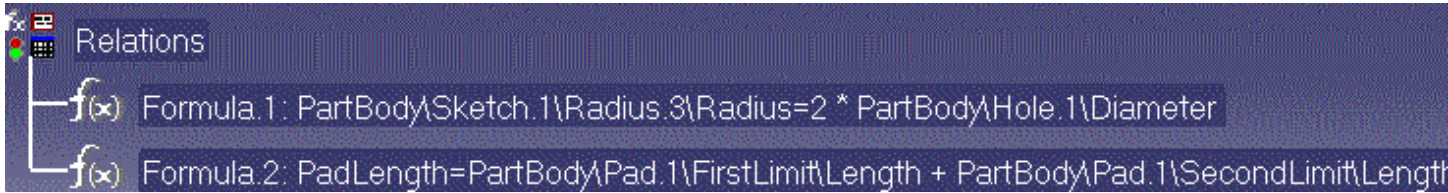


3. Use the 'Wizard' button to help you type the  $2 * \text{PartBody\Hole.1\Diameter}$  relation by selecting items from the dictionary list.



- Click OK in the "Formula Editor" once your relation is typed. The Formula.1 relation is added to the specification tree. In the parameter list of the dialog box:
  - a formula is now associated with the pad radius
  - the new Relations\Formula.1\Activity parameter is added.
- In the parameter list, select the PadLength item, click 'Add Formula' to create the formula below:  

$$\text{PadLength} = \text{PartBody}\backslash\text{Pad.1}\backslash\text{FirstLimit}\backslash\text{Length} + \text{PartBody}\backslash\text{Pad.1}\backslash\text{SecondLimit}\backslash\text{Length}$$
 In the parameter list, the Formula.2 relation is now associated with the PadLength user parameter. In the specification tree, PadLength is also displayed with the value resulting from Formula.2. This is now what you should see in the specification tree under "Relations":




- Click OK in the "Formulas" dialog box to terminate this task. **Keep your document open and proceed to the next task.**

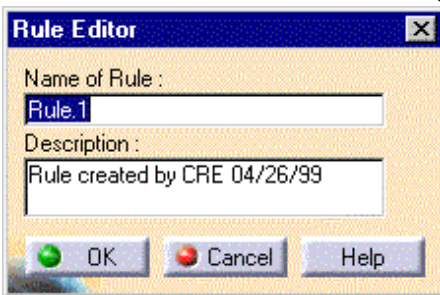


## Rules

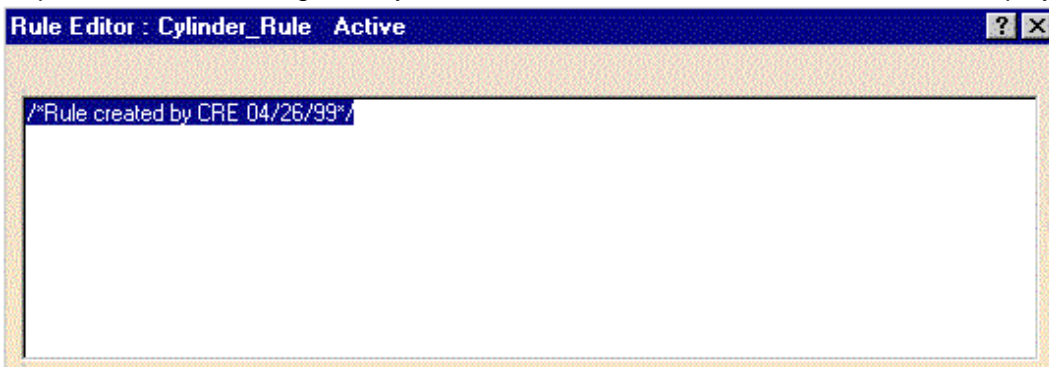
Unlike the parameter and formula capabilities which are available to all CATIA users, the rule and check capabilities require the Knowledge Advisor product. This task is a way to introduce the Knowledge Advisor rules without getting bogged down in details. For more information on rule-related tasks, refer to [Rules](#).



- Select the KwrStartDocument item in the specification tree, then access the Knowledge Advisor workbench (select Knowledge Advisor workbench from the Start->Infrastructure menu).
- Click the  rule icon. The following dialog box is displayed:



- Replace the Rule.1 string with Cylinder\_Rule and click OK. The "Rule Editor" is displayed.



4. Type the code below into the edition box or copy/paste it from your browser to the edition box.

```
PartBody\Hole.1\Activity = true
if PadLength <= 50mm and PadLength > 20mm
{
PartBody\Hole.1\Diameter = 20mm
Message("PadLength is: # | Internal Diameter is: #",
PadLength,PartBody\Hole.1\Diameter)
}
else if PadLength > 50mm and PadLength < 100mm
{
PartBody\Hole.1\Diameter = 50mm
Message("PadLength is: # | Internal Diameter is: #",
PadLength,PartBody\Hole.1\Diameter)
}
else if PadLength >= 100mm
{
PartBody\Hole.1\Diameter = 80mm
Message("PadLength is: # | Internal Diameter is: #",
PadLength,PartBody\Hole.1\Diameter)
}
else
{
PartBody\Hole.1\Activity = false
Message("PadLength is: # | Internal Diameter is: #",
PadLength,PartBody\Hole.1\Diameter)
}
}
```


5. Click Apply. An information window displays the PadLength and Pad internal diameter values. Click OK in the Information window. The Cylinder\_Rule relation is added to the specification tree.
6. Click OK to terminate this part of the dialog. **Keep your document open and proceed to the next task.**

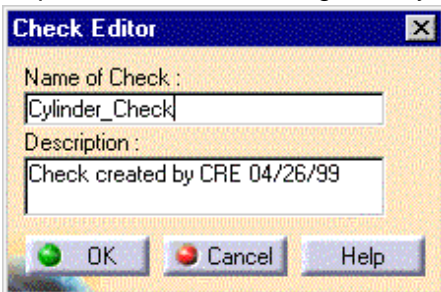


## Checks

The Knowledge Advisor product is required. Refer to [Checks](#) for more information on checks and check-related tasks.



1. Click the  icon. The first "Check Editor" dialog box is displayed.
2. Replace the Check.1 string with Cylinder\_Check,



then click OK. The "Check Editor" box is displayed. It is similar to the Rule Editor.

3. Select the Information item in the 'Type of Check' list.
4. Enter a string in the message field (for example: Pad too short). This message is to be displayed whenever the statement specified by the check is not fulfilled.
5. Enter the following statement into the edition box:

```
PadLength > 20mm
```

6. Click Apply. The Cylinder\_Check relation is added to the specification tree. A green icon in the specification tree means that the check is fulfilled. No message is displayed.
7. Click OK to confirm the check creation.
8. Change the Pad limits so that PadLength <= 20 mm. You should be warned by an information window that the check

is no longer valid.

The resulting document is KwrForRuCh.CATPart.



# Basic Tasks

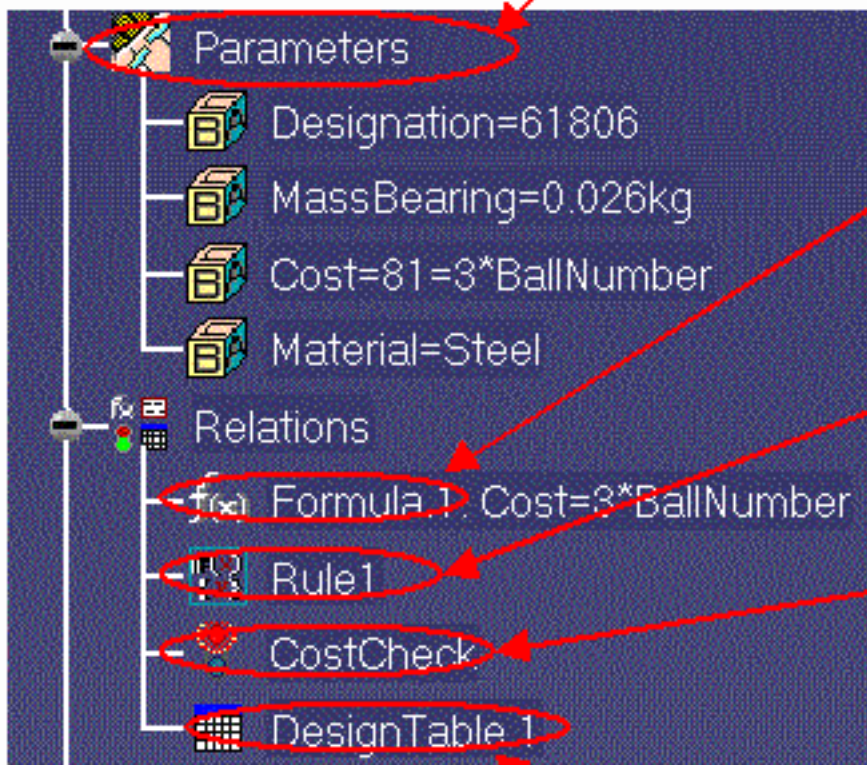
Here are the Knowledge Advisor features. Click the hyperlinks on figure below to display the list of interactive tasks whereby you can create and manipulate these features. The principles behind these features are explained in:

- [About Parameters](#)
- [About Formulas](#)
- [About Rules and Checks.](#)

The [Knowledge Inspector](#) is a tool to study impacts and dependencies. It uses as input data the knowledgware features described below.

In figure below, click any of the links to display the related summary of tasks.

  [Parameter](#) Tasks



 [Formula](#) Tasks

 [Rule](#) Tasks

 [Check](#) Tasks

 [Design Table](#) [Advanced Tasks](#)



The functions related to parameters and formulas are provided to all CATIA users. They are part of the CATIA Infrastructure capabilities. But you will make the best use of them in a Knowledge Advisor application.

The functions related to rules and checks as well as the Knowledge Inspector require the Knowledge Advisor product.

# About Parameters

When you create a part like the hollow cylinder of our "Getting Started" example, you often start by creating a sketch, then you create a pad by extruding the initial sketch, then you add other features to the pad created. The final document is made up of features which define the intrinsic properties of the document. Removing one of these features results in a modification of the document. These features are called **parameters**. Parameters play a prominent role in knowledgeware applications. They are features that can be constrained by relations and they can also be used as the arguments of a relation.

In addition to these parameters, CATIA allows you to create **user parameters**. These user parameters are extra pieces of information added to a document.

## User Parameter Types

User parameters can be of various types:

- Real
- Integer
- String
- Boolean
- Length
- Angle
- Time
- Mass
- Volume
- Density
- Area
- Moment of inertia
- Energy
- Force
- Inertia
- Mass flow rate
- Moment
- Pressure
- Angular stiffness
- Temperature
- Linear mass
- Linear stiffness
- Volumetric flow rate
- Frequency
- Electric power
- Voltage
- Electric resistance
- Electric intensity

User parameters are very handy in knowledgeware applications:

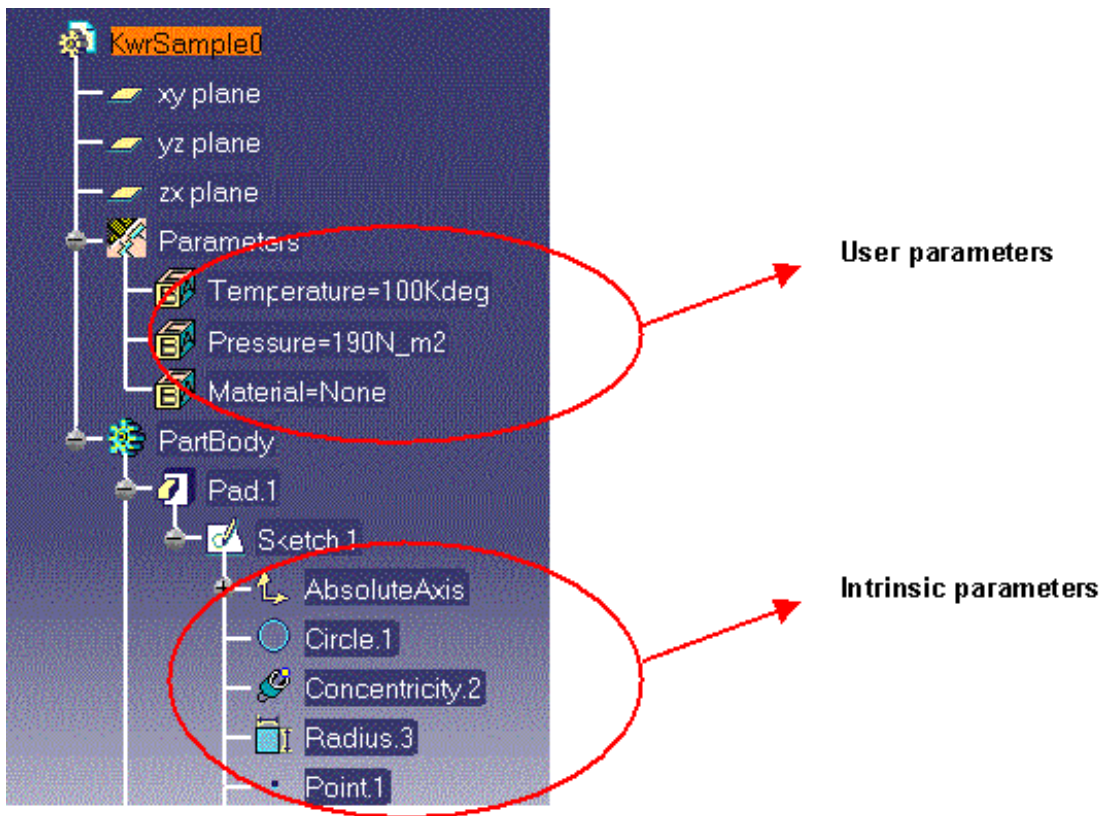
- they can be used to add specific information to a document
- they can be defined or constrained by relations
- they can be used as the arguments of a relation.

A given relation may take as its arguments both types of parameters (intrinsic and user).

 User parameters can be assigned single or **multiple values**.

## Displaying Parameters in the Specification Tree

The user parameters are displayed in the specification tree only if the Tools->Options->Part->Display->Parameters box is checked. The user parameter list contains at least the Material parameter. Its initial value is set to None.



The user parameter values are displayed in the specification tree only if the Tools->Option>Knowledge->Parameter Tree View->With Value box is checked.

### Copy/Pasting Parameters

The Tools->Options->Knowledge->Pasting Parameters check boxes allow you to:

- paste a parameter without the formula which defines it.  
For example: Holeplus= 15 = Diameter + 10 will be pasted as Real.i = 15 (if the With Value box is checked)
- paste a parameter as well as the formula which defines it, but only if the parameters referred to in the formula are also selected in the copy.  
For example: Holeplus= 15 = Diameter + 10 will be pasted as Real.i = 15 if the Diameter parameter does not belong to the items selected for the copy but HolePlus will be pasted as Real.i = 15 = Real.j + 10 if Diameter is selected in the copy (use multi-selection).
- paste a parameter as well as the formula.  
Holeplus= 15 = Diameter + 10 will be pasted as Real.i = Diameter + 10

### Parameters and National Support Languages

CATIA users working with non-latin characters should check the Tools->Options>Knowledge->Parameter Names->Surrounded by ' option. Otherwise, parameter names should have to be renamed in latin characters when used in formulas.

### Importing User Parameters

User parameters can be imported from an external file. This external file can be either an Excel file(on Windows NT) or a tabulated text file. Here are examples of import files in an Excel format and in a tabulated text format .

	A	B	C	D
1	Force	100N ; <50N> ; 150N		
2	Area	<5m2> ; 20m2		
3	Pressure	0N_m2 ;	Force/Area	Maximum pressure allowed
4				

```
T:\CompanyFile0 - Notepad
File Edit Search Help
Force 100N ; <50N> ; 150N
Area <5m2> ; 20m2
Pressure 0N_m2 ; Force/Area Maximum pressure allowed
```

Here are the formatting rules to be applied:

- Column 1  
Parameter names
- Column 2  
Parameter values. *Multiple values are allowed.* Values should then be separate by a ";". The imported value is the one delimited by the "<" and ">" tags. Use the Tab key to skip from one column to the other in a tabulated text file.
- Column 3  
Formula. If no formula is specified, the third column should be left empty. In a tabulated text file, just press the Tab key twice from column 2 to leave column 3 empty.
- Column 4  
Optional comment.



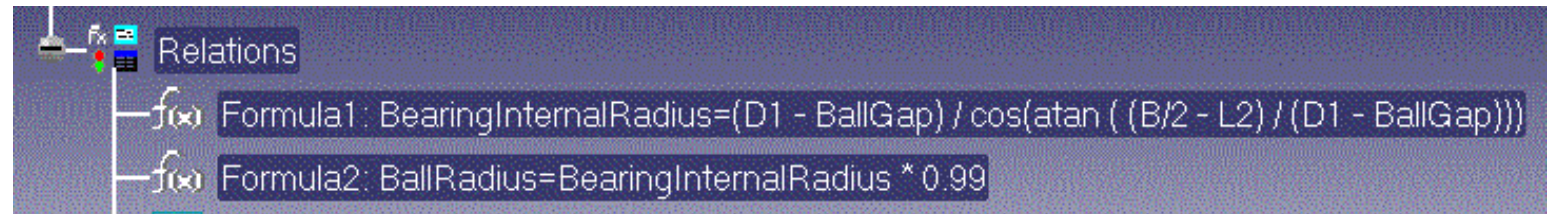
# About Formulas

Formulas are features used to define or constrain a parameter. A formula is a relation that you write with parameters, operators and functions. The left part of the relation is the parameter to be constrained, the right part is a statement.

Once it has been created, a formula can be manipulated like any other feature from its contextual menu.

## Displaying Formulas in the Specification Tree

Formulas are displayed in the specification tree only if the Tools->Options->Part->Display->Relations box is checked.





In addition, whenever a user parameter is constrained by a formula, the formula is displayed with the parameter it constrains if the Tools->Options->General->Knowledge->Parameter Tree View->With Formula box is checked.



## The Activity Parameter

A formula is a feature which is assigned a parameter called the *activity*. The activity value is of boolean type and is defined as follows:

Activity value	false	true
Parameter value	Does not depend on the formula	Is calculated from the formula
Relation icon in the specification tree		

If a formula is created for a parameter which not already constrained by another formula, the activity of the new formula is set to true by default. A parameter can be constrained by several formulas, but only one formula can be active at a time. Before activating a formula on a given parameter, you must deactivate the other formulas defined on the same parameter.

## Importing Formulas

Parameters as well as the associated formulas can be imported from an external file. Refer to [About Parameters](#) and [Importing Parameters](#) for more information on how to import formulas.

 About Formulas

 Creating a Formula

 Deleting a Formula

 Activating and Deactivating :

 Importing Formulas

 Writing Formulas

# About Rules and Checks

Rules and Checks are relations that can only be created and managed with the Knowledge Advisor product.

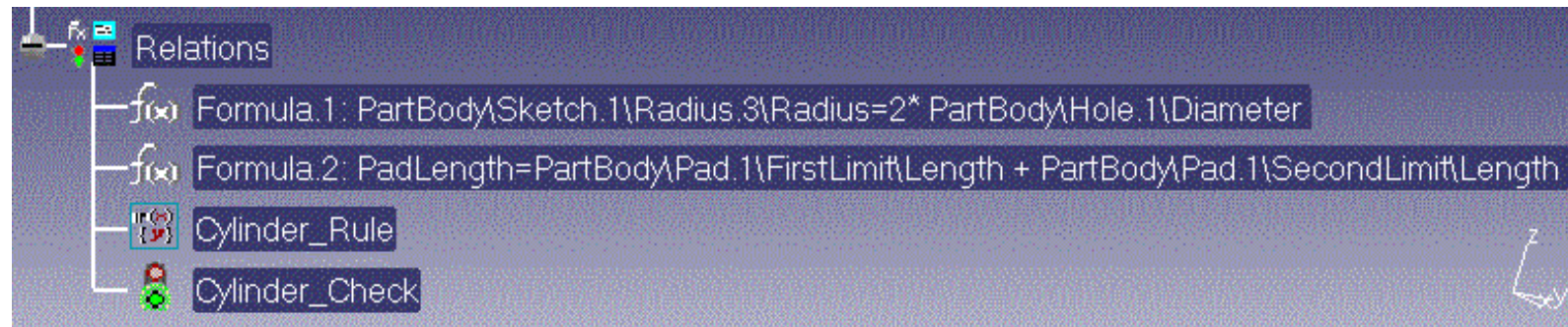
A rule is a set of instructions whereby you can control certain parameters and events according to a context. A check is a set of instructions whereby you are warned that some statements are fulfilled or not.

Once they have been created, rules and checks can be manipulated like any other features:

- Double-clicking on a rule or on a check in the specification tree displays an edition window
- Delete, Cut/Copy/Paste and Rename operations can be accessed from the contextual menu.

## Displaying Rules and Checks in the Specification Tree

Rules and Checks are displayed in the specification tree only if the Tools->Options->Part->Display->Relations box is checked.



## The Rule Activity Parameter

Like formulas, rules are assigned a parameter called the *activity*. The activity value is a boolean defined as follows:

Activity value	false <i>the rule is not executed</i>	true <i>the rule is executed</i>
Relation icon in the specification tree		

## The Check Parameters

A check is assigned three parameters defined as follows:

	Activity	Severity	Result
Value	<ul style="list-style-type: none"> <li>• true the check is executed  or </li> <li>• false the check is not executed  or </li> </ul>	<ul style="list-style-type: none"> <li>• Silent See <a href="#">Creating a Check</a></li> <li>• Information See <a href="#">Creating a Check</a></li> <li>• Warning See <a href="#">Creating a Check</a></li> </ul>	<ul style="list-style-type: none"> <li>• true the statements specified by the check are fulfilled </li> <li>• false the statements specified by the check are not fulfilled </li> </ul>



# Using the Knowledge Inspector

The Knowledge Inspector allows you to query a design to determine and preview the results of changing any parameters without committing themselves to actually changing the design. This "what if" analysis provides immediate feedback that helps you experiment and refine designs.

While it is important to be able to determine what happens when one or more parameters are changed, it is equally significant for you to be able to see how a design can be changed to achieve a desired result. The Knowledge Inspector supports this by allowing you to query "how to" make a particular change.

In brief, the Knowledge Inspector is a tool to study impacts and dependencies.

<b>What if</b> ( <i>impacts</i> )	Helps you understand to what extent changing any parameter of your design (such as material, pressure, or a dimensional parameter) changes the operation or design of the product on which you are working. Can be used to examine interactions of parameters with each other and with the rules that make up the product's specifications. A "Geometric Update" option enables you to visualize the result of your modification in the geometry area.
<b>How To</b> ( <i>dependencies</i> )	Helps you determine how your design can be changed to achieve a desired result.

 You shouldn't use the  capabilities with Knowledge Inspector.



# Parameters

## [Create a user parameter](#)



Select a type in the 'New Parameter of type' list, then click 'New Parameter of type'.

## [Edit/modify a user parameter](#)



Select the parameter to be edited in the "Formulas" dialog box, then modify its value in 'Edit name, value or formula' .

or

In the specification tree, double-click the parameter to be edited, then modify its value in the "Edit Parameter" editor.

or

In the specification tree, right-click the parameter to be edited, then select the *Parameter.object->Definition* function from the contextual menu.

## [Add a comment to a user parameter](#)



In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula', then select 'Edit Comment...' from the contextual menu.

or

Whatever the edition box, right-click the value field(s), then select 'Edit comment...' from the contextual menu.

## [Add a tolerance to a magnitude type parameter](#)



In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula', then select 'Add Tolerance' from the contextual menu.

or

Whatever the edition box, right-click the value field(s), then select 'Add Tolerance...' from the contextual menu.

## [Specify lower and upper bounds](#)



In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula', then select 'Add Range' from the contextual menu.

Whatever the edition box, right-click the value field(s), then select 'Add Range...' from the contextual menu.


### Specify an increment/decrement amount




In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula', then select 'Change Step' from the contextual menu.

or  
Whatever the edition box, right-click the value field(s), then select 'Change Step...' from the contextual menu.

### Specify the Material parameter value

Select the Material parameter in the specification tree, then click .

or  
Double-click the Material parameter in the specification tree, then enter the material value in the dialog box.

or  
 In the "Formulas" dialog box, select the Material parameter in the parameter list, then modify its value in 'Edit name, value or formula'.

### Update the Mechanical Property parameters

Select the Properties->Mass option from the feature contextual menu and click OK.

### Import parameters



In the "Formulas" dialog box, click `Import . . .`. A file selection dialog box is displayed. Select either a .xls file(Windows NT only) or a .txt file.

### Delete a parameter



Select the parameter to be deleted in the "Formulas" dialog box, then click 'Delete Parameter'.

or  
In the specification tree, right-click the parameter to be deleted, then select the Delete function from the contextual menu.




# Creating and Editing a Parameter

A document is defined by features called *parameters*. When you create a document, you define interactively a number of parameters. In order to add information to a document, you can create new parameters that can be used later on when defining formulas, rules or checks (for those of you using the Knowledge Advisor). Refer to [About Parameters](#) for a further information.

Here, we are going to create a time parameter (corresponding for example to the time required to machine the part).



1. Open the KwrStartDocument.CATPart document.
2. Click the Formula icon . The "Formulas" dialog box is displayed.
3. Select the Time item with 'Single Value' in the 'New Parameter of type' list, then click 'New Parameter of type'. The new parameter appears in 'Edit name, value or formula'.  
Note that a parameter can be assigned multiple values. You just have to select the 'Multiple values' option at parameter creation and enter the values one-by-one to specify the value list.
4. Replace the `Time.1` name with `Machining_Time` and assign the 1000s value to this parameter. Then click Apply. The `Machining_Time` parameter is added to the specification tree. The dialog box is modified as follows:

Parameter	Value
PartBody\Hole.1\ThreadingDepth	20mm
PartBody\Hole.1\Activity	true
Mechanical_Property\Volume	0m3
Mechanical_Property\Mass	0kg
Mechanical_Property\WetArea	0m2
Material	None
<b>Machining_Time</b>	<b>1000s</b>

Edit name, value or formula

Machining\_Time 1000s =

New Parameter of type Time With Single Value

5. Click OK to terminate the dialog.
6. To edit the parameter you have just created, you can:
  - either, double-click on the `Machining_Time` parameter in the specification tree. This will display an "Edit Parameter" box that you can use to modify your parameter value.
  - or, in the specification tree, select the `Machining_Time.object->Definition` function from the contextual menu.

- or, in the "Formulas" dialog box, select the Machining\_Time parameter, then modify its value in 'Edit name, value or formula'.

The resulting document is KwrParam0.CATPart.



The parameter list displayed in the "Formulas" dialog box depends:

- on the current feature. For example, if you select Pad.1 in the KwrStartDocument.CATPart specification tree, the list will contain only the Pad.1 parameters. If you select the document global feature by clicking KwrStartDocument, all the document parameters will be displayed in the list.
- on the option selected for the Filter field of the "Formulas" dialog box. Setting the All option displays all the parameters of the current feature. Setting the User option displays only the parameters you have added to your document with the 'New Parameter of type' button.

When editing a parameter, a number of functions can be accessed from the contextual menu of the value field(s). These functions allow you to add attributes to the parameter and to manipulate the formula which constrains the parameter (when this formula exists). In addition, when a parameter value is constrained by a formula, the "Edit Parameter" dialog box displays a formula icon that allows you to edit the formula. When a parameter value is driven from a design table, the "Edit Parameter" dialog box displays a design table icon that allows you to edit the design table.

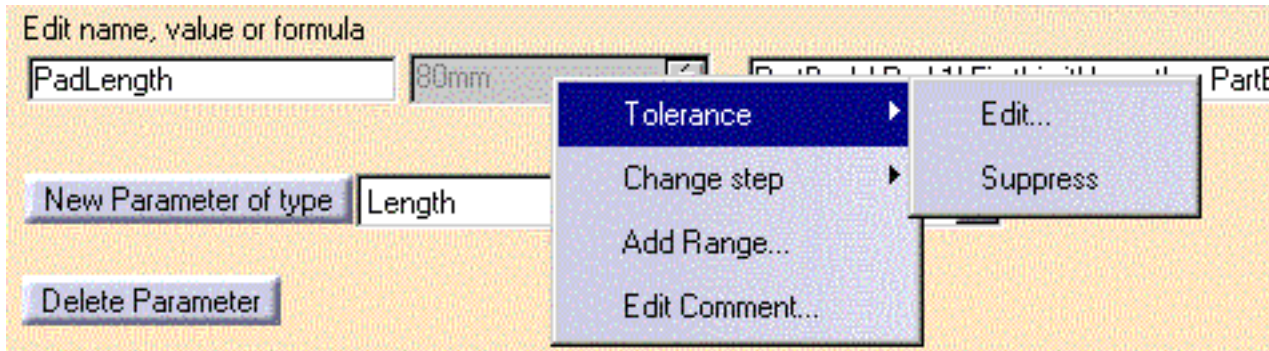
To display parameter values in the geometry area, you must have the "Formulas" dialog box open and select a feature in the specification tree. Selecting a parameter in the parameter list will highlight this parameter in the geometry area.



# Parameter Comments and other Parameter Attributes

To enrich the definition of a parameter, you can add some attributes to it. The attributes you can add to a parameter depend on the parameter type. These attributes can be specified in two ways:

1. either from the contextual menu of any edition box. For example, if you edit a pad, the contextual menu of the value fields provides you with functions to add parameter attributes
2. or from the 'Edit name, value or formula' contextual menu of the "Formulas" dialog box.



The second method is described below.

## Adding Comments

Comments can be added to all parameter types.



1. In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula'.
2. Select the Edit Comment... function from the contextual menu
3. Enter your comments in the dialog box which is displayed.



## Adding a Tolerance

The tolerance attribute can only be added to magnitude type parameters.



1. In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula'.
2. Select the Tolerance function from the contextual menu. Then select Edit. A dialog box is displayed. The default values are those specified in the Tools->Options->Tolerance tab for the Length and Angle parameters.
3. Enter the maximum and minimum tolerance values.

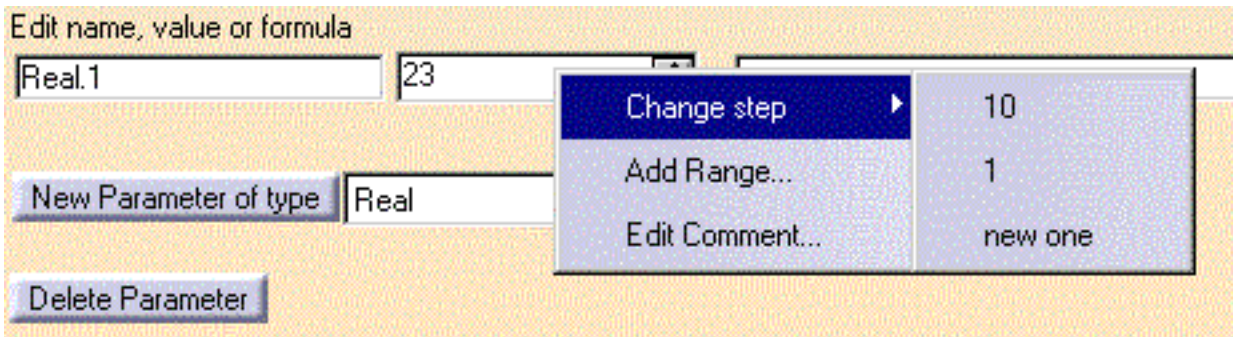


## Modifying the Increment/Decrement Amount

In the parameter value field of 'Edit name, value or formula', the arrow buttons can be used to increment or decrement values by a certain amount. You can modify the amount being incremented or decremented.



1. In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula'.
2. Select the 'Change step' function from the contextual menu.
3. Select one of the amount displayed.
4. If the new amount you want to apply to the value field is not displayed, select 'new one' and enter the new amount in the box which is displayed.



5. Click Enter to apply the new amount to the value field.

Note that, in addition to the initial increment/decrement amount (1 in the figure above) and to the 'new one' option, the 'Change step' option displays the list of increment/decrement amounts which have already been declared.



## Specifying Lower and Upper Bounds

Specifying lower and upper bounds provides you with a means to be warned whenever a parameter value is out of range.



1. Select the 'Add Range' function from the contextual menu.
2. In the dialog box which is displayed, enter the upper and lower bound values.
3. Click OK. If the current parameter value is out of range, a message is displayed. CATIA proposes to replace the current value with one of the bounds you have specified.

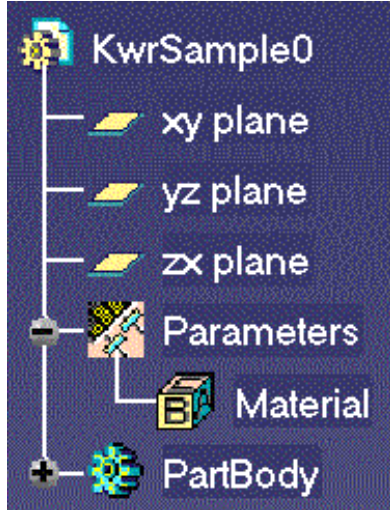


# Specifying the Material Parameter Value

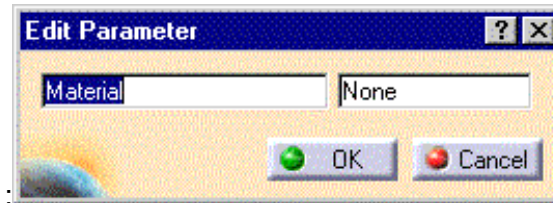
Whatever your document, the Material parameter is always displayed in the specification tree. By default, its value is set to None. The Mechanical\_Property features are calculated from the Material value. You need to specify a material to set the values of the Mechanical\_Property features.




1. Open the KwrStartDocument.CATPart document.  
The Material parameter appears by default in the specification tree. Its value is set to None.




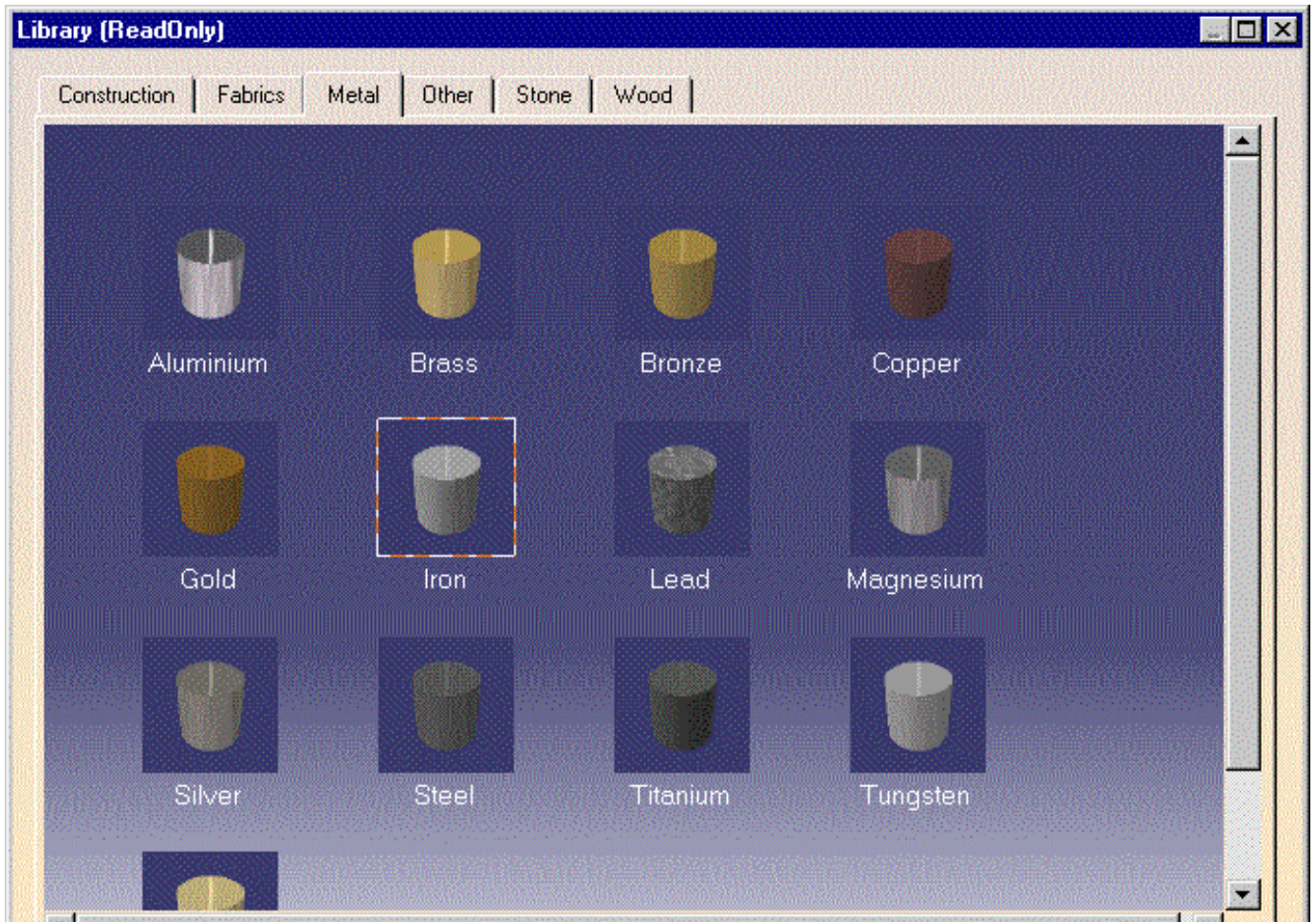
Double-clicking the Material feature in the specification tree displays the following dialog box

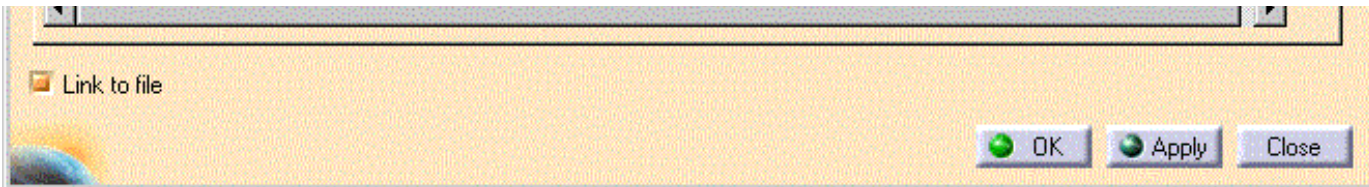


Here are the values displayed in the parameter list when clicking .

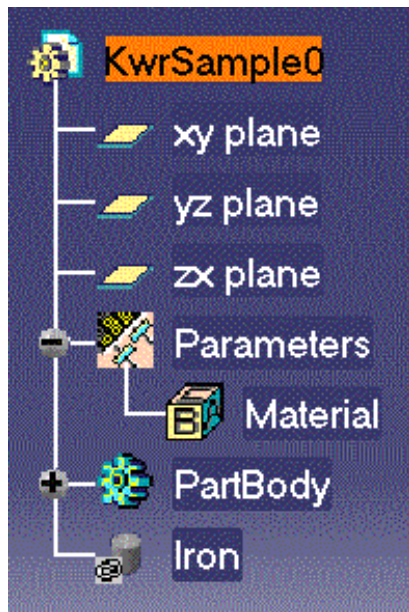
Mechanical_Property\Volume	0m3
Mechanical_Property\Mass	0kg
Mechanical_Property\WetArea	0m2
Material	None

2. Click the KwrStartDocument feature in the specification tree.
3. Click the  icon in the standard toolbar to display the available material library.
4. Select the Metal->Iron material.

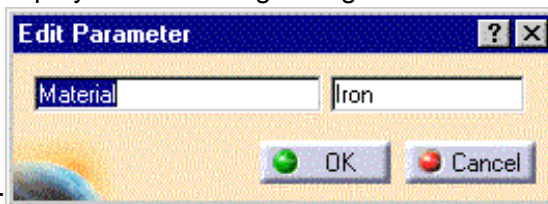





5. Click Apply. The Iron feature is added to the specification tree.



Now, double-clicking the Material feature in the specification tree displays the following dialog box



And here are the values displayed in the parameter list when clicking .

Mechanical_Property\Volume	0m3
Mechanical_Property\Mass	0kg
Mechanical_Property\WetArea	0m2
Material	Iron

6. Click OK in the "Edit Parameter" and "Formulas" dialog boxes.


7. Keep your document open and go to [Valuating the Mechanical Property Parameters.](#)



# Updating the Mechanical\_Property Parameters

Once the Material value has been specified, the Mechanical\_Property parameters are automatically updated **when the Properties option is selected in the contextual menu**. This is what you have to do to display the updated Volume, Mass and WetArea properties after you have assigned a value to the Material parameter.



1. Open the Properties dialog box from the contextual menu.
2. Select the Mass tab. The document mechanical properties have been updated from the value assigned to the Material parameter.
3. Click OK to go back to your document.
4. Click the Formula icon . In the parameter list, the mechanical property values are updated.

Mechanical_Property\Volume	2.19911e-005...
Mechanical_Property\Mass	0.158336kg
Mechanical_Property\WetArea	0.00989602m2
Material	Iron


5. Click OK to terminate the dialog. The resulting document is KwrMaterial0.CATPart.



# Importing Parameters

Parameters and parameter values can be imported from a text file or from an Excel file (Windows™ NT). See [About Parameters](#) for more information on the import file format.



1. Open the KwrStartDocument.CATPart.
2. Click the Formula icon . The "Formulas" dialog box is displayed.
3. Click **Import . . .**. A file selection dialog box is displayed.
4. Select either the ExCompanyFile0.xls file(Windows NT only) or the TxCompanyFile0.txt file. Then click **Open**. The list of parameters to be imported into the KwrStartDocument.CATPart document is displayed.

Parameters and formulas created by the import operation

Name	Value	Formula	Comment
Force	50N		
Area	5m2		
Pressure	10N_m2	Force/Area	Maximum pressure all..

OK Cancel

5. Click **OK** to import all the parameters from the input file to the KwrStartDocument.CATPart document. The imported parameters are now displayed in the parameter list of the "Formulas" dialog box and in the specification tree.

Parameter	Value	Formula
Mechanical_Property\Volume	0m3	
Mechanical_Property\Mass	0kg	
Mechanical_Property\WetArea	0m2	
Material	None	
Force	50N	
Area	5m2	
Pressure	10N_m2	

6. Click **OK** to terminate the dialog. The resulting document is KwrlImport.CATPart.



# Deleting a Parameter

The only parameters that you can delete in a knowledgeware application are those created by the 'New Parameter of type' button.

You can delete a parameter:




1. either from the contextual menu. **The Material parameter cannot be deleted.**
2. or by using the 'Delete Parameter' button of the "Formulas" dialog box. To do so:
  1. In the Parameter list of the "Formulas" dialog box, select the parameter to delete
  2. Click `Delete Parameter`. A message prompts you to confirm the deletion.
  3. Click `Yes`. The parameter deleted should no longer appears in the specification tree.

In the KwrParam0.CATPart document, the only parameter you can delete is the `Machining_Time` parameter.




# Formulas



## Create a formula

-  In the "Formulas" dialog box, select the parameter to be constrained, then click 'Add Formula'. Enter the formula in the "Formula Editor".
- or  In the "Formulas" dialog box, select the parameter to be constrained, then enter the formula in 'Edit name, value or formula'.
- or  In the "Formulas" dialog box, double-click the parameter to be constrained and enter the formula in the "Formula Editor".
- or When editing a parameter, right-click the value field(s), then select Formula->Edit from the contextual menu.

## Edit/modify a formula

-  In the parameter list of the "Formulas" dialog box, select the parameter defined by the formula, then modify the formula in 'Edit name, value or formula'.
- or In the specification tree, right-click the formula to be edited, then select the Formula.object->Definition function from the contextual menu.
- or In the specification tree, double-click the formula to be edited.
- or In the specification tree, double-click the user parameter, then click the  $f(x)$  button in the "Edit Parameter" dialog box.
- or When editing a parameter, right-click the value field(s), then select Formula->Edit from the contextual menu.

## Activate/deactivate a formula

-  In the parameter list of the "Formulas" dialog box, select the parameter which is constrained by the formula to (de)activate and use the Activate/Deactivate check box right of the 'Edit name, value or formula' field.
- or  In the parameter list of the "Formulas" dialog box, select the formula/activity parameter and modify its value in 'Edit name, value or formula'.

- or In the specification tree, right-click the formula whose activity is to be modified, then select the *Formula.object*->(De)activate function from the contextual menu.
- or When editing a parameter constrained by a formula, right-click the value field(s), then select the Formula->(De)activate function from the contextual menu.

### Import parameters and formulas



In the "Formulas" dialog box, click 'Import', then specify the import file path. The import file should be either a .txt file or a .xls file.

### Delete a formula



In the parameter list of the "Formulas" dialog box, select the parameter which is constrained by the formula to be deleted, then click 'Delete Formula'.

- or In the specification tree, right-click the formula to be deleted, then select the Delete function from the contextual menu.
- or When editing a parameter constrained by a formula, right-click the value field(s), then select the Formula->Delete function from the contextual menu.

# Creating and Editing a Formula




CATIA Version 5 allows you to define a document parameter with respect to other parameters. The relation which defines how a parameter value is calculated from other parameter values is called a *formula*. This task explains how to create a formula specifying that the external radius of a hollow cylinder is twice its internal diameter.

Make sure the Relations option is active in the Tools -> Options -> Part -> Display tab.

Note that the radius of a sketch can be defined by a formula provided it is declared as a constraint



1. Open the KwrStartDocument.CATPart.

2. Click the Formula icon . The "Formulas" dialog box is displayed.

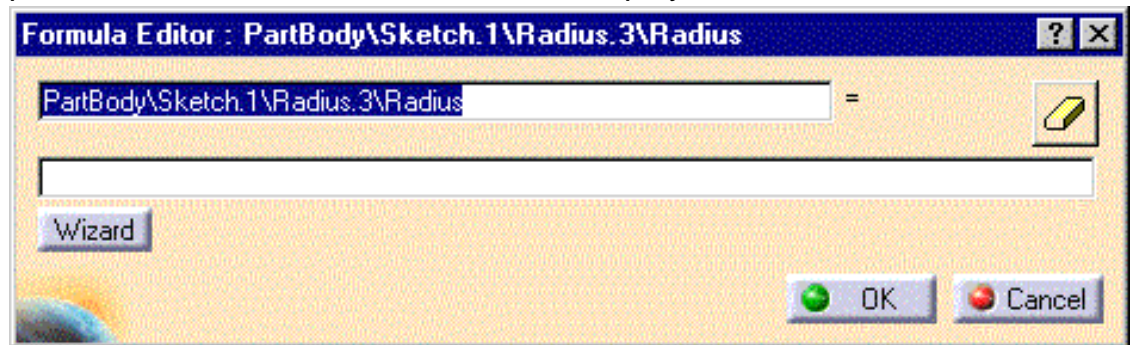
3. Add the formula to the pad radius parameter. To do this, you can follow one of the three methods described below:

## **Method 1**

1. Select the PartBody\Sketch.1\Radius.3\Radius in the parameter list.
2. Enter directly the  $2 * \text{PartBody}\backslash\text{Hole.1}\backslash\text{Diameter}$  relation in the right part of the 'Edit name, value or formula' field. Go to [Writing Formulas](#) for tips and tricks for writing formulas.
3. Press the Tab key or the Enter Key.

## **Method 2**

1. Double-click the PartBody\Sketch.1\Radius.3\Radius parameter in the parameter list. The "Formula Editor" is displayed.




2. Enter the  $2 * \text{PartBody}\backslash\text{Hole.1}\backslash\text{Diameter}$  relation in the formula field. Go to [Writing Formulas](#) for information on how to use the Wizard.


If needed, use the  icon to erase a formula.

3. Click OK in the "Formula Editor".

## **Method 3**

1. Select the PartBody\Sketch.1\Radius.3\Radius in the parameter list.
2. Click 'Add Formula'. The "Formula Editor" is displayed.
3. Enter the  $2 * \text{PartBody}\backslash\text{Hole.1}\backslash\text{Diameter}$  relation in the formula field. If needed, use the  icon to erase a formula. Go to [Writing Formulas](#) for tips and tricks for writing formulas.
4. Click OK in the "Formula Editor".

At this stage of the dialog:

- the Formula.1 relation is added to the specification tree.
- in the "Formulas" dialog box:
  - the formula is added to the parameter list
  - in the parameter list, the PartBody\Sketch.1\Radius.3\Radius parameter is now defined by a formula
  - in the 'Edit name, value or formula field', the parameter value is now grayed out. The formula appears in the right part of the field.
- In the geometry area, a parameter which is constrained by a formula is indicated by an  icon.

4. Click Apply to update the document in the geometry area.

5. Click OK to terminate the dialog. The resulting document is KwrFormula0.CATPart.



 About Formulas

 Creating a Formula

 Deleting a Formula

 Activating and Deactivating :

 Importing Formulas

 Writing Formulas

# Activating and Deactivating a Formula

A parameter value can be defined by a number of formulas. But only one formula can be active at a time.


By default, a formula is active at creation but, if a formula defines a parameter which is already constrained by another formula, you will be asked whether the new formula:

- should be activated and the other one deactivated (if previously activated)
- should be deactivated
- should be cancelled (no formula created).

A formula can be activated or deactivated: either from the "Formulas" dialog box, or from the contextual menu, or by a rule execution.

Here is a scenario illustrating how to activate and deactivate a formula.



1. Open the KwrFormula1.CATPart. Two formulas are added to the document. One defines the PartBody\Sketch.1\Radius.3\Radius parameter with respect the hole diameter, the other defines the hole diameter with respect to the Pad limits. Both formulas are active.
2. Click the Formula icon , select the PartBody\Sketch.1\Radius.3\Radius parameter, then click 'Add Formula'.
3. Enter the PartBody\Sketch.1\Radius.3\Radius = 4 \* PartBody\Hole.1\Diameter relation in the "Formula Editor".
4. Click Yes. As the Pad radius is already constrained by the Formula.1 relation, a dialog box asks you:
  - either to deactivate Formula.1 and to create Formula.3 as active
  - or to keep Formula.1 active and to create Formula.3 as inactive
  - or to cancel the Formula.3 creation.
5. Click Yes to create the Formula.3 relation as active and deactivate the Formula.1 relation. Deactivate Formula.3 by using one of the three methods below:

## Method 1

Set the Relations\Formula.3\Activity value to false

Relations\Formula.1\Activity	false
Relations\Formula.2\Activity	true
Relations\Formula.3\Activity	false
Material	None

Edit name, value or formula

Relations\Formula.3\Activity false =

## Method 2

Use the Activate/Deactivate check box

Double click on a parameter to edit it

Parameter	Value	Formula	Active
PartBody\Sketch.1\Radius.3\Radius	160mm	4 * PartBody\Hole.1\Diameter 2 * PartBody\Hole.1\Diameter	no
PartBody\Pad.1\FirstLimit\Length	40mm		
PartBody\Pad.1\SecondLimit\Length	0mm		
PartBody\Hole.1\HoleLimit.1\Depth	40mm		
PartBody\Hole.1\HoleLimit.1\Angle	120deg		
PartBody\Hole.1\Diameter	40mm	PartBody\Pad.1\FirstLimit\Lengt...	yes

Edit name, value or formula

PartBody\Sketch.1\Radius. 160mm = 4 \* PartBody\Hole.1\Diameter

Activate/Deactivate check box

## Method 3

1. Click OK to exit the "Formulas" dialog box.
2. Click the Formula.3 relation in the specification tree.
3. Select the Formula.3 object -> Deactivate option from the contextual menu.

6. Re-activate Formula.1 by using one of the three methods above.
7. Click OK to terminate the dialog. The resulting document is KwrFormula2.CATPart.



 About Formulas

 Creating a Formula

 Deleting a Formula

 Activating and Deactivating :

 Importing Formulas

 Writing Formulas

# Importing Formulas

Formulas are imported with the parameters they define. See [About Parameters](#) for more information on the import file format.



When a parameter is defined by a formula, units should be mentioned in column 2.

Examples of import files with formulas are provided in the ExCompanyFile0.xls and TxCompanyFile0.txt tables.



# Deleting a Formula

You can delete a formula in three ways:

1. In the specification tree, right-click the formula to be deleted, then select the Delete function from the contextual menu.
2. In the parameter list of the "Formulas" dialog box, select the parameter constrained by the formula to be deleted, then click 'Delete Formula'. A message asks you to confirm the deletion. Click Yes. The formula deleted should no longer appears in the specification tree.
3. When editing a parameter constrained by a formula, right-click the value field and select the Formula->Delete function from the contextual menu.



# Rules

## Create a rule



Enter the rule name and comments. Then write your rule in the rule editor. To enter a parameter within a rule, you can:

- either click the parameter in the specification tree
- or click the dimension displayed in the geometry area
- or use the editor dictionary.

## Edit/modify a rule

In the specification tree, double-click(twice) the rule to be edited.

or

In the specification tree, right-click the rule to be edited, then select the *Rule.object*->Edition function from the contextual menu.

## Activate/deactivate a rule

In the specification tree, right-click the rule to be activated or deactivated, then select the *Rule.object*->(De)activate function from the contextual menu.

or



In the parameter list of the "Formulas" dialog box, select the rule/activity parameter modify its value in 'Edit name, value or formula'.

## Delete a rule

In the specification tree, right-click the rule, then select the Delete function from the contextual menu.

# Checks

## Create a check



Enter the check name and comments. Then write your check in the check editor. To enter a parameter within a check, you can:

- either click the parameter in the specification tree
- or click the dimension displayed in the geometry area
- or use the Dictionary.

### Edit/modify a check

In the specification tree, double-click the check to be edited.

or

In the specification tree, right-click the check to be modified, then select the *Check.object->Edition* function from the contextual menu.

### Activate/deactivate a check

In the specification tree, select the check to be activated or deactivated, then select the *Check.object->(De)activate* function from the contextual menu.

or



In the parameter list of the "Formulas" dialog box, select the check/activity parameter and modify its value in 'Edit name, value or formula'.


### Delete a check

In the specification tree, right-click the check to be deleted, then select the *Delete* function from the contextual menu.

# Creating and Editing a Rule

A rule is a set of instructions whereby you can control certain parameters and events according to a context.

To create a rule:

1. open your document
2. access the Knowledge Advisor workbench
3. click the Rule icon . The Rule Editor dialog box is displayed. Now it is mainly a programming task. Refer to [Programming Rules](#).

When creating rules, remember that, if a rule is intended to constrain a parameter, other relations constraining this parameter will conflict with the rule to be created. To resolve the conflict, CATIA will ask you to specify which relations are to be declared active or inactive.

To edit a rule from the specification tree, you can:

- either double-click the rule
- or right-click the rule and select the Rule.Object->Definition function from the contextual menu.



# Programming Rules

The Knowledge Advisor script language provides you with a full range of operators, functions and keywords that can be used to program your rules.

Rules are a **multiple-line** pieces of code that you can write either by typing directly the appropriate syntax in the editor field or by selecting items from the editor dictionary list. Here are some details about the rule syntax. Other dictionary components are described in [Writing Formulas](#).

## Tips and Tricks for Editing Rules

In the rule editor, to enter a parameter, you can:

- either click it in the specification tree
- or click the dimension displayed in the geometry area
- or use the dictionary

By default, the dictionary displays the list of all the parameters defining your document. But this list can be restricted to a single-type parameter list. For example, if you select the Length item in the "Members of Parameters" list, only the Length type parameters will be displayed.

## Using Temporary Variables

Temporary variables can be declared by using the **let** keyword. A temporary variable does not persist as a parameter after the rule execution is finished.

```
/*Rule created by CRE 08/23/99*/  
let x = 5 mm  
if PartBody\Hole.1\Diameter > x  
{  
PartBody\Hole.1\Activity = false  
}
```

Temporary variables should be declared at the beginning of the rule, before any conditional instruction is specified.

## Including Comments

The **/\*** and **\*/** comment characters are supported.

```

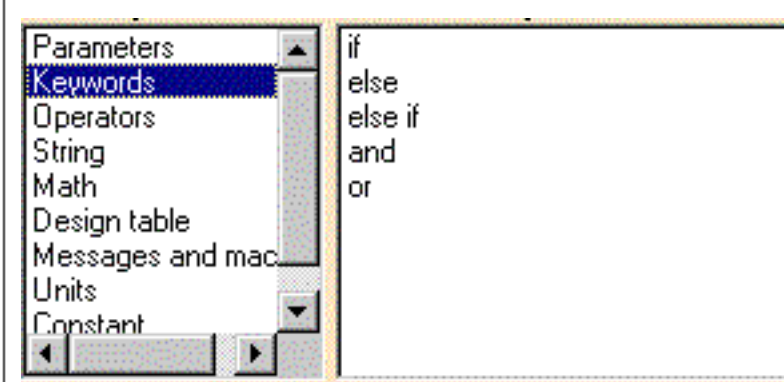
/* Rule created by CRE 05/03/99 */
if PartBody\Sketch.1\Radius.3\Radius > 45mm
{
    LaunchMacroFromFile("Macro1.CATScript")
}
else
/*
    LaunchMacroFromFile("Macro2.CATScript")
*/
Message("No macro launched")

```

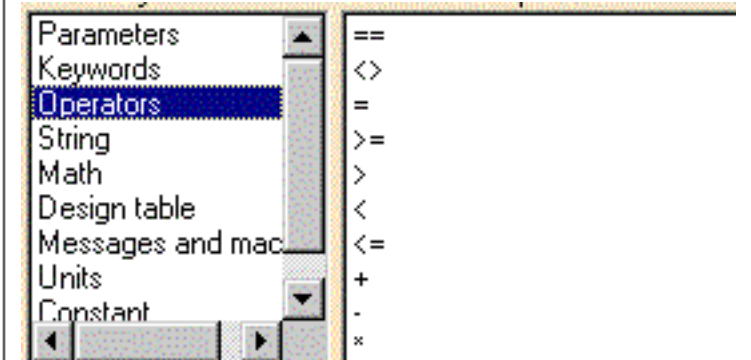
## The Rule Dictionary

The dictionary components which are not already described in [Writing Formulas](#) are discussed below.

### Conditional Statements or Keywords

	<p>Like in many languages, the <b>if....else...else if</b> statements are used to evaluate whether a condition is <b>true</b> or <b>false</b> and, depending on the result, to specify one or more statements to run. Usually the condition is an expression that uses a comparison or logical operator to compare one value or parameter value with another. Statements within a condition can be combined using the <b>and</b> or <b>or</b> keywords.</p>
---	---

### Operators

	<p>Operators provided are:</p> <ul style="list-style-type: none"> <li>● arithmetical operators</li> <li>● and comparison operators. Note that comparison operators should be only used in conditional statements. <b>They can't be used as functions as they don't return any value.</b></li> </ul>
--	---

### Message and Macro Functions

Parameters  
Keywords  
Operators  
String  
Math  
Design table  
Messages and mac  
Units  
Constant

Message  
LaunchMacroFromFile

The Rule editor provides you with the functions listed on the figure opposite:

- [Message](#)
- [LaunchMacroFromFile](#).



Up



About Rules and Checks



Creating and Editing a Rule



Creating and Editing a Check



Programming Checks



Programming Rules

# Writing Formulas

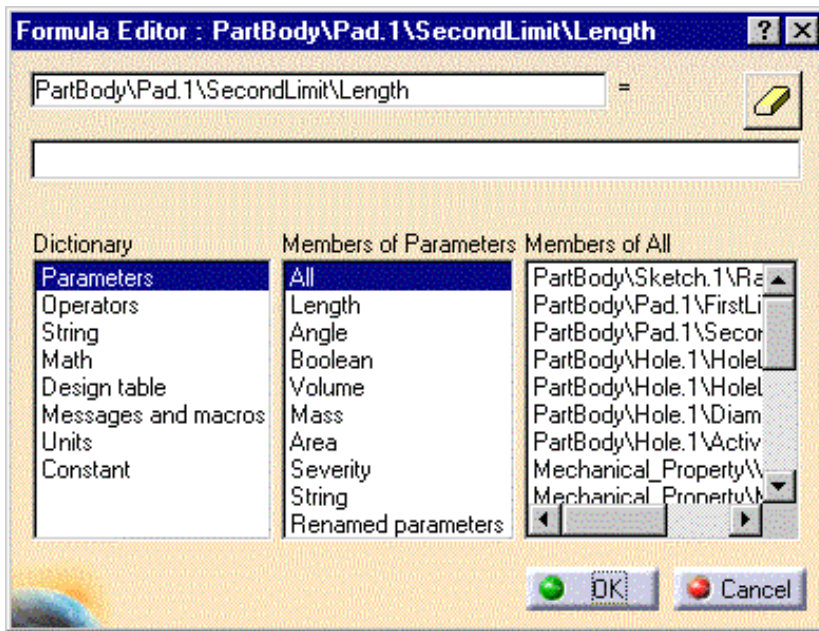
The formula editor provides you with a full range of operators and functions that you can use with parameters to create formulas. A formula is a **one-line** statement that you can write either by typing directly the appropriate syntax in the editor field or by selecting items from the editor dictionary list. The formula syntax is easy to use and learn. Here are some details about how to use the formula editor.

## Tips and Tricks for Editing Formulas


In the "Formula Editor", to enter a parameter, you can:

- either click it in the specification tree
- or click the dimension displayed in the geometry area
- or use the Wizard.

To use the Wizard, press the "Wizard" button in the "Formula Editor". The dialog box which is displayed looks something like this.



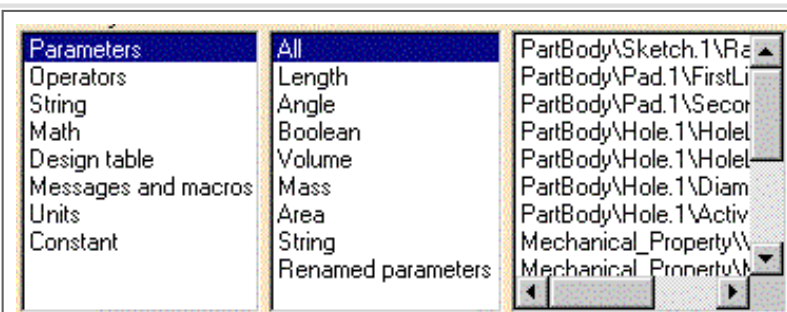
By default, the dictionary displays the list of all the parameters defining your document. But this list can be restricted to a single-type parameter list. For example, if you select the Length item in the "Members of Parameters" list, only the Length type parameters will be displayed.

Use the  to erase a formula.

## The Formula Dictionary

Here is a description of the language provided by the formula dictionary.

### Parameters

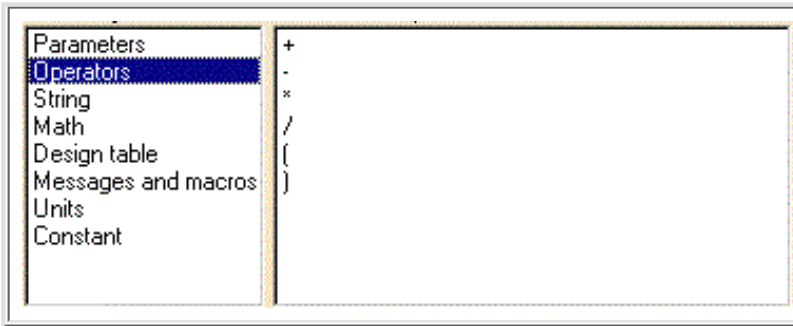


When a parameter name contains a special character, it should be delimited by back-quotes when used within a formula.

No extra-space should be left after the first and before the last back-quotes.

Example:  
Pad-2\MechanicalPartProperties\Density

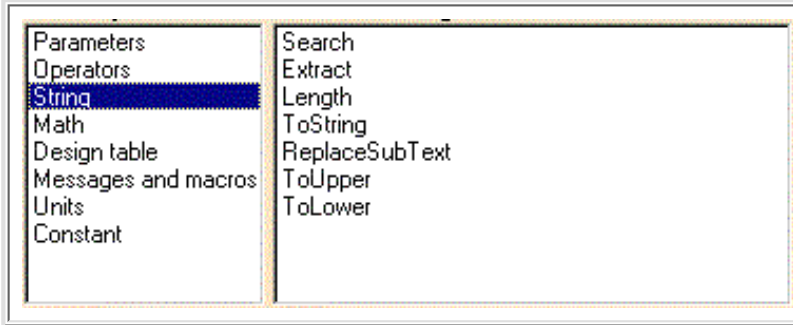
### Arithmetic Operators



The formula editor provides you with the operators listed on the figure opposite.

Note that the "+" operator can be used for string concatenation.

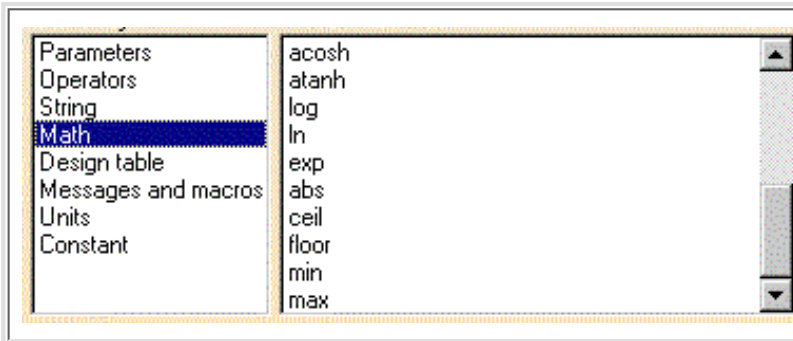
## String Functions



Here are the functions provided to manipulate strings:

- [Search](#)
- [Extract](#)
- [Length](#)
- [ToString](#)
- [ReplaceSubText](#)
- [ToUpper](#)
- [ToLower.](#)

## Mathematical Functions

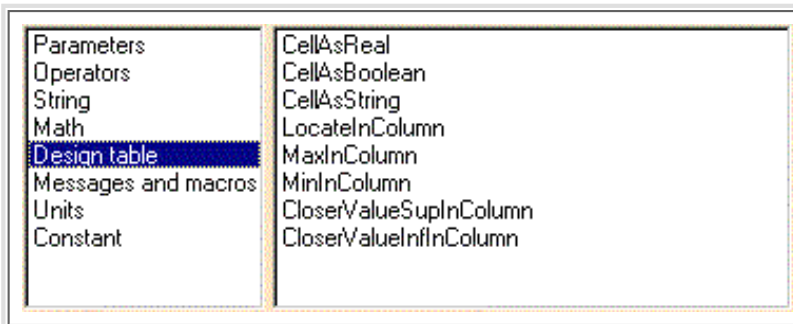


The full range of trigonometric functions is provided to you as well as some other usual functions.

**Note** `floor` returns the largest integer value that is less than or equal to the value specified in the argument.

`ceiling` returns the smallest integer value that is greater than or equal to the value specified in the argument.

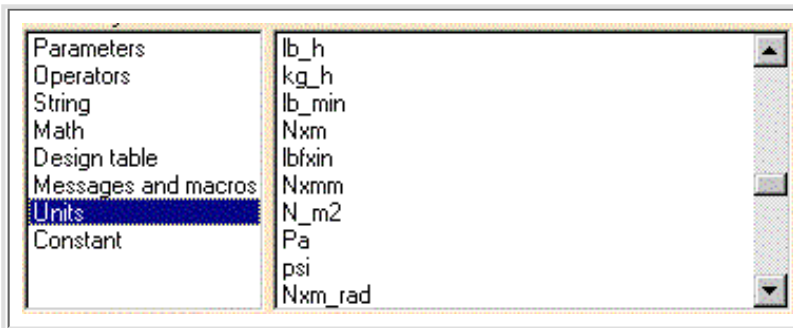
## Design Table Functions



Here are the functions to access the design table:

- [CellAsReal](#)
- [CellAsBoolean](#)
- [CellAsString](#)
- [LocateInColumn](#)
- [MaxInColumn](#)
- [MinInColumn](#)
- [CloserValueSupInColumn](#)
- [CloserValueInfInColumn](#)

## Units



1. Pay attention to unit consistency when writing a formula.
2. When importing formulas, don't omit to specify a value (unit included) in column 2.
3. Units are written with an underscore instead of the usual "/" (example N\_m2 instead N/m2).

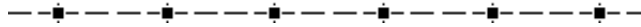
## Constants

The following constants are specified or recognized by CATIA when programming formulas. As a result, they can be used anywhere in a formula in place of the actual values.

- false - one of the two values that a parameter of type Boolean can have
- true - one of the two values that a parameter of type Boolean can have
- PI - **3.14159265358979323846** - The ratio of the circumference of a circle to its diameter.
- E - The base of natural logarithm - The constant e is approximately **2.718282**.



# Search Function



## Description

Searches for the first occurrence of a substring in a String type parameter. Returns the index of the start of the substring. Returns -1 if the substring specified is not found.

## Syntax

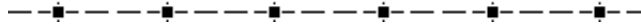
### **Search**(String)

The **Search** function has one argument:

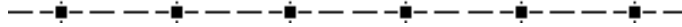
<i>Argument</i>	<i>Description</i>
<i>String</i>	Required. Name of the substring to be searched for (should be put in quotes) or name of the String parameter.

## Example

```
Responsible = Cilas Evans = ...  
I1Search = 6 = Responsible.Search("EVANS")  
I2Search = 0 = Responsible.Search("Cilas")  
I3Search = -1 = Responsible.Search("CILAS")
```



# Extract Function



## Description

Returns the substring starting at a given position with a specified length.

## Syntax

**Extract**(*StartIndex*, *Length*)

The **Extract** function has two arguments:

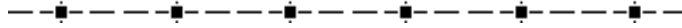
<i>Arguments</i>	<i>Description</i>
<i>StarIndex</i>	Required. Index of the substring first character
<i>Length</i>	Required. Substring length

## Formula Example

FirstName = Cilas  
Name = Cilas = Responsible.Extract(0,FirstName.Length())  
Extract = la = FirstName.Extract(2,2)



# Length Function



## Description

Returns the string length.

## Syntax

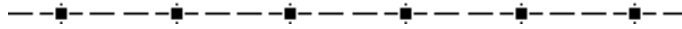
### **Length()**

The **Length** function has no arguments

## Formula Example

FirstName = Cilas

Name = Cilas = Responsible.Extract(0,FirstName.Length())



# ToString Function

---

## Description

Converts an integer into a string.

## Syntax

**ToString**(*Integer*)

The **ToString** function has one argument:

<i>Argument</i>	<i>Description</i>
<i>Integer</i>	Required. Integer to be converted into string.

## Formula Example

RevNumber = 1

Revision = Rev: 1 = "Rev: " + ToString(RevNumber)

---

# ReplaceSubText Function

---

## Description

Replaces a substring with another substring within a character string.

## Syntax

**ReplaceSubText**(*InputString*,*SubString1*,*SubString2*)

The **ReplaceSubText** function has three arguments:

<i>Argument</i>	<i>Description</i>
<i>InputString</i>	Required. Input string.
<i>SubString1</i>	Required. Substring to be replaced
<i>SubString2</i>	Required. Substring replacing SubString1

## Formula Example

Responsible = Cilas EVANS

NewResponsible = Eazy EVANS = ReplaceSubText(Responsible,"Cilas","Eazy")

---

# ToUpper Function

---

## Description

Changes all lower-case letters of a string to upper-case.

## Syntax

**ToUpper**(*String*)

The **ToUpper** function has one argument:

<i>Argument</i>	<i>Description</i>
<i>String</i>	Required. Name of the string to be converted to upper-case (should be put in quotes) or name of the String parameter.

## Formula Example

FamilyName = Evans

FirstName = Cilas

Responsible = Cilas EVANS = FirstName + " " + ToUpper(FamilyName)

---

# ToLower Function

---

## Description

Changes all upper-case letters of a string to lower-case.

## Syntax

**ToLower**(*String*)

The **ToLower** function has one argument:

<i>Argument</i>	<i>Description</i>
<i>String</i>	Required. Name of the string to be converted to lower-case (should be put in quotes) or name of the String parameter.

---

## Description

Returns the contents of a cell located in a column intended for real values. Returns zero if the cell does not contain a real or if the function parameters are not properly specified (see remarks).

## Syntax

**CellAsReal**(*DesignTableName*, *RowNumber*, *ColumnNumber*)

The **CellAsReal** function has three arguments:

<i>Arguments</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Name of the design table you want to read. This name is the string which identifies the design table in the specification tree. Should be put in quotes.
<i>RowNumber</i>	Required. Line or configuration number. Integer from 1 to n.
<i>ColumnNumber</i>	Required. Number or index of the table column. Integer from 1 to n.

## Remarks

The real which is returned by the function should be added to your document as a user parameter.

The **CellAsReal** function returns a nil value:

- when the design table passed as its first argument does not exist
- when the row or column numbers are out of range or not specified as integers.

# CellAsBoolean Function

[Code sample](#)  
[Input design table](#)

## Description

Returns the contents of a cell located in a column intended for boolean values. Returns false if the cell does not contain a boolean or if the function parameters are not properly specified (see remarks).

## Syntax

**CellAsBoolean**(*DesignTableName*, *RowNumber*, *ColumnNumber*)

The **CellAsBoolean** function has three arguments:

<i>Arguments</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Name of the design table you want to read. This name is the string which identifies the design table. Should be put in quotes.
<i>RowNumber</i>	Required. Line or configuration number. Integer from 1 to n.
<i>ColumnNumber</i>	Required. Number or index of the table column. Integer from 1 to n.

## Remarks

The boolean which is returned by the function should be added to your document as a user parameter.

The **CellAsBoolean** function returns a false value:

- when the design table passed as its first argument does not exist
- when the row or column numbers are out of range or not specified as integers.

# CellAsString Function

[Code sample](#)  
[Input design table](#)

## Description

Returns the contents of a cell located in a column. Returns an empty string if the cell is empty or if the function parameters are not properly specified (see remarks).

## Syntax

**CellAsString**(*DesignTableName*, *RowNumber*, *ColumnNumber*)

The **CellAsString** function has three arguments:

<i>Arguments</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Name of the design table you want to read. This name is the string which identifies the design table in the specification tree. Should be put in quotes.
<i>RowNumber</i>	Required. Line or configuration number. Integer from 1 to n.
<i>ColumnNumber</i>	Required. Number or index of the table column. Integer from 1 to n.

## Remarks

The string which is returned by the function should be added to your document as a user parameter.

The **CellAsString** function returns an empty string:

- when the design table passed as its first argument does not exist
- when the row or column numbers are out of range or not specified as integers.

# LocateInColumn Function

[Code sample](#)  
[Input design table](#)

## Description

Returns the index of the first row which contains a specified value. Returns zero if the value is not found or if the function arguments are not properly specified (see remarks).

## Syntax

**LocateInColumn**(*DesignTableName*, *ColumnNumber*, *Value*)

The **LocateInColumn** function has three arguments:

<i>Arguments</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Name of the design table you want to read. This name is the string which identifies the design table in the specification tree. Should be put in quotes.
<i>ColumnNumber</i>	Required. Number or index of the table column. Integer from 1 to n.
<i>Value</i>	Required. Value searched for. Can be a string or a boolean

## Remarks

The **LocateInColumn** function returns 0:

- when the design table passed as its first argument does not exist
- when the row or column numbers are out of range or not specified as integers.

# MaxInColumn Function

---

## Description

Returns the greatest of a column values. Returns 0 if the column does not contain numerical values or if the function arguments are not properly specified (see remarks).

## Syntax

**MaxInColumn**(*DesignTableName*, *ColumnNumber*)

The **MaxInColumn** function has two arguments:

<i>Arguments</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Name of the design table you want to read. This name is the string which identifies the design table in the specification tree. Should be put in quotes.
<i>ColumnNumber</i>	Required. Number or index of the table column. Integer from 1 to n.

## Remarks

The **MaxInColumn** function returns 0:

- when the design table passed as its first argument does not exist
  - when the row or column numbers are out of range or not specified as integers.
-

# MinInColumn Function

---

## Description

Returns the smallest of a column values. Returns 0 if the column does not contain numerical values or if the function arguments are not properly specified (see remarks).

## Syntax

**MinInColumn**(*DesignTableName*, *ColumnNumber*)

The **MinInColumn** function has two arguments:

<i>Arguments</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Name of the design table you want to read. This name is the string which identifies the design table in the specification tree. Should be put in quotes.
<i>ColumnNumber</i>	Required. Number or index of the table column. Integer from 1 to n.

## Remarks

The **MinInColumn** function returns 0:

- when the design table passed as its first argument does not exist
  - when the row or column numbers are out of range or not specified as integers.
-

# CloserValueSupInColumn Function

[Code sample](#)  
[Input design table](#)

## Description

Scans the values of a column and returns the greatest cell value which is the nearest to a specified one. Returns 0 if no value is found or if the function arguments are not properly specified (see remarks).

## Syntax

**CloserValueSupInColumn**(*DesignTableName*, *ColumnNumber*, *RealValue*)

The **CloserValueSupInColumn** function has three arguments:

<i>Arguments</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Name of the design table you want to read. Should be put in quotes.
<i>ColumnNumber</i>	Required. Number or index of the table column. Integer from 1 to n.
<i>RealValue</i>	Required. Value searched for. Should be a real.

## Remarks

The **CloserValueSupInColumn** function returns 0:

- when the design table passed as its first argument does not exist
- when the row or column numbers are out of range or not specified as integers.

# CloserValueInfInColumn Function

[Code sample](#)  
[Input design table](#)

## Description

Scans the values of a column and returns the smallest cell value which is the nearest to a specified one. Returns 0 if no value is found or if the function arguments are not properly specified (see remarks).

## Syntax

**CloserValueInfInColumn**(*DesignTableName*, *ColumnNumber*, *RealValue*)

The **CloserValueInfInColumn** function has three arguments:

<i>Arguments</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Name of the design table you want to read. This name is the string which identifies the design table in the specification tree. Should be put in quotes.
<i>ColumnNumber</i>	Required. Number or index of the table column. Integer from 1 to n.
<i>RealValue</i>	Required. Value searched for. Should be a real.





## Remarks

The **CloserValueInfInColumn** function returns 0:

- when the design table passed as its first argument does not exist
- when the row or column numbers are out of range or not specified as integers.


# Creating and Editing a Check

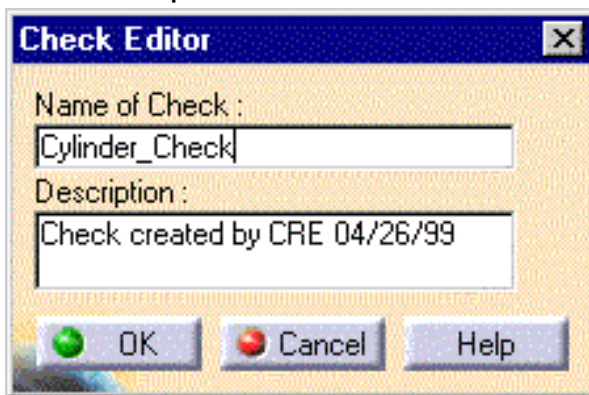
Like a rule, a check is a piece of code. But unlike a rule, it has no impact on parameter values. When executed, a check just gives you a clue as to whether some conditions are verified or not. There are three types of checks: the silent, the information and the warning checks. Depending on the type of check and the result of the check, you will be warned as follows:

	Check verified	Check not verified
Relation icon in the specification tree		
<b>Silent</b> check	no message displayed	no message displayed
<b>Information</b> check	no message displayed	 the message specified at check creation is displayed in an information box
<b>Warning</b> check	no message displayed	 the message specified at check creation is displayed in a warning box.

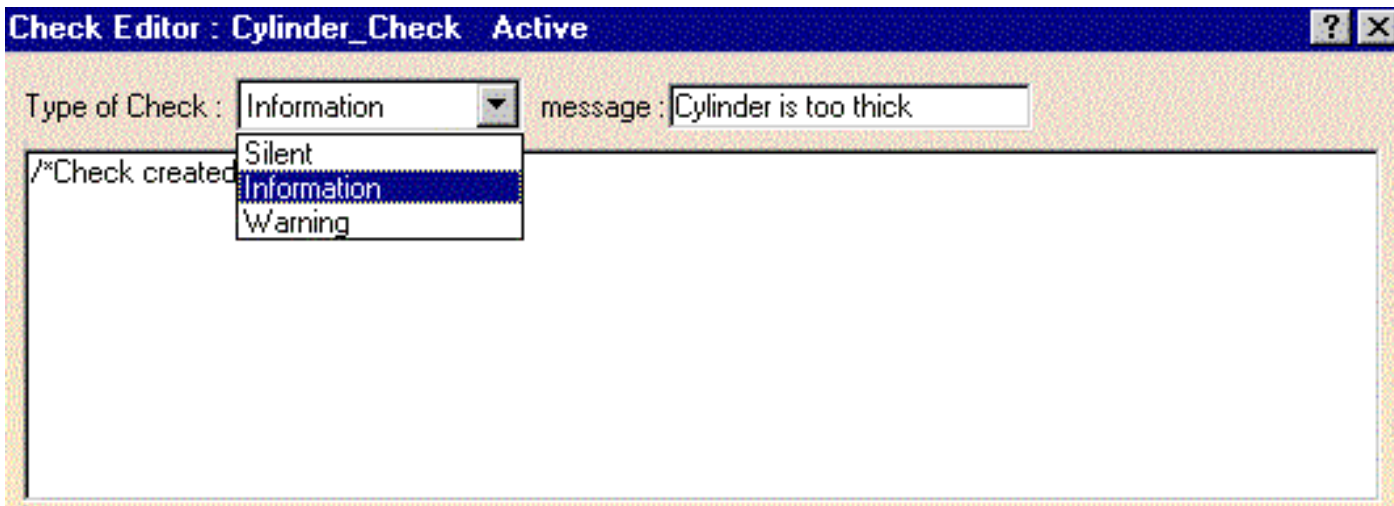
Creating a check is partly a programming task. But the check syntax is easy to learn. Refer to [About Rules and Checks](#) and [Programming Checks](#).



1. Open the KwrFormula0.CATPart document and access the Knowledge Advisor workbench.
2. Click the Check icon . The first Check Editor dialog box is displayed.
3. Replace the default name with Cylinder\_Check. If needed, add some comments into the Description field.



4. Click OK. The Check Editor is displayed.



5. Select a type of check. Enter the message you want to be displayed in the information or warning box in case the check is not verified.
6. Enter the check statements in the edition window. You can simply Copy/Paste the following statements into the edition window:
 

```
if PartBody\Sketch.1\Radius.3\Radius - PartBody\Hole.1\Diameter / 2 > 10mm
then
Relations\Formula.1\Activity == false
```
7. Click Apply to test your check syntax as well as the result on the document. The Cylinder\_Check is added to the relations in the specification tree. Depending on whether it is verified or not, the light icon associated with the check relation is either green or red.
8. Click OK to terminate the dialog.
9. To edit the check you have just created from the specification tree, you can:
  - double-click the check
  - right-click the check and select the Check.object->Definition function from the contextual menu.

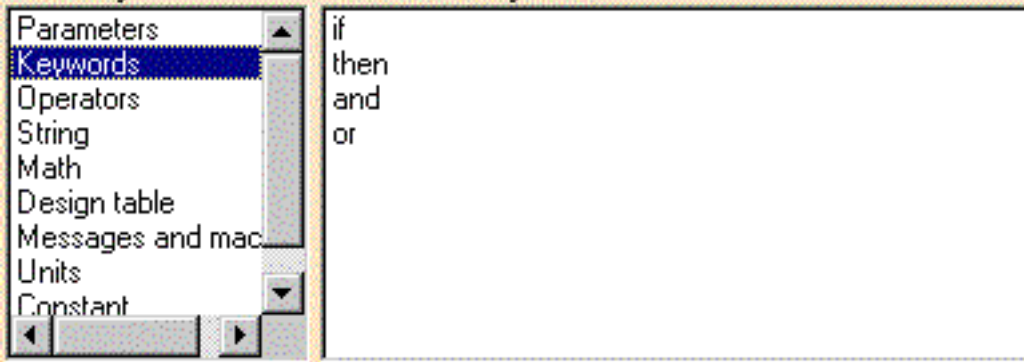


# Programming Checks

The Knowledge Advisor script language provides you with a full range of operators, functions and keywords that can be used to program your checks.

Checks are a **multiple-line** pieces of code that you can write either by typing directly the appropriate syntax in the editor field or by selecting items from the editor dictionary list. Here are some details about the check syntax. Other dictionary components are described in [Writing Formulas](#) and [Programming Rules](#).

## Conditional Statements or Keywords



Like in many languages, the **if....then...** statements are used to evaluate whether a condition is fulfilled or not and, depending on the result, to specify one or more statements to verify. Usually the condition is an expression that uses a comparison or logical operator to compare one value or parameter value with another. Statements within a condition can be combined using the **and** or **or** keywords.



# Advanced Tasks

Among the knowledgware capabilities, there are two which are not very basic at all. The objects you manipulate are a bit complicated and, in addition to interactive tasks, you will often have to perform programming tasks. Two features describe these knowledgware capabilities :

1.  ***the design table***

A design table is a tool mainly intended to ease the definition of mechanical parts. It is provided to all CATIA users. But you will make the best use of it in a Knowledge Advisor application. A design table can be created from a CATIA document, the document data is then exported to the design table. It can also be applied to a document, the document data is then imported from the design table.

The Design Table tasks and concepts are described in the Infrastructure documentation. Refer to [About Design Tables](#) for further information on the design table principles. Go to [Design Table Tasks](#) for a summary of interactive tasks as well as a comprehensive list of programming functions.

2.  ***the behavior***

A behavior is a set of VB Script instructions which are executed when a feature responds to a given event. Go to [Behavior Tasks](#) for information on this very particular feature.

# About Design Tables

A design table provides you with a means to create and manage component families. These components can be for example mechanical parts just differing in their parameter values. Screws are a good example of mechanical parts that can be described by a design table. To simplify, imagine they all described by four parameters: the head width, the head height, the body width and the body height. The sets of four parameter values that can be assigned to a screw can be easily regrouped in a design table. This design table has as many columns as screw parameters and as many rows as sets of parameter values. In a design table, a set of parameter values is called a *configuration* and it is registered in a row.

A design table is a tool mainly intended to ease the definition of mechanical parts. It is provided to all CATIA users. But you will make the best use of it in a Knowledge Advisor application. A design table can be created from a CATIA document, the document data is then exported to the design table. It can also be applied to a document, the document data is then imported from the design table.

The purpose of the design table is to drive the parameters of a CATIA document from external values. These values are stored in the form of a table either in a Microsoft® Excel file on Windows™ or in a tabulated text file. When using a design table the trick is to associate the right document parameters with the right table parameters. The design table columns may not all correspond to your document parameters and you may decide to apply only part of the design table values to your document. By creating *associations*, you declare what document parameters you want to link with what table columns.

The design table becomes a more powerful tool when it is used with the Knowledge Advisor. You are provided with functions to read the design table parameters. These design table functions can be used when programming your checks and rules. Using these functions spares you all the association operations.

## The Excel Sheet Format (on Windows)

The values mentioned in the sheet cells have to be expressed in appropriate units. Otherwise, the right values won't be associated with the document parameters.

If no unit is mentioned within a cell:

- the unit taken into account is the one mentioned in the first row
- and if no unit is specified in the first row, the unit taken into account is the relevant MKS system unit.

Here is an example of an Excel sheet:

Annotations:

- column name
- column unit
- When a configuration which contains empty cells is selected, the parameters associated with the empty cells are not modified. This property enables you to modify parameters but only under certain conditions
- Within a given column, you can change the units
- Units can be specified in cells. If no unit is mentioned, the parameter evaluation is carried out...

	A	B	C	D	E	F	G	H	I
1	Designation	d(mm)	D(mm)	B(mm)	d1(mm)	D1(mm)	MinFiletRadius(mm)	BearingMass	Material
2	623	1.5	5	4	2.6	3.75	0.15	1.5g	Aluminium
3	618/4	2	4.5	2.5	2.6	3.75	0.15	0.7g	Iron
4	624	2	6.5	5		5.15	0.2	3.1g	Gass
5	634	2	8	5	4.2	6	0.3	5.4g	Aluminium
6	618/5	2.5	5.5	3	3.4	4.6	0.15	1.2	Iron
7	61800	5	9.5	5	6.3	8.2	0.3	5.5	Pavement
8	6000	5	13	8	7.2	10.7	0.3	19	Italian Marble
9	61802	7.5	12	5		10.55	0.3	7.4	Iron
10	6302	7.5	21	13	11.85	16.95	1	82	Pavement
11	61804	10	16	7	12	14.15	0.3	18	Aluminium
12	6404	10	36	19	18.55	27.8	1.1	400	Gass
13	61806	15	21	7	16.9	19.1	0.3	26	Pavement
14	6306	15	36	19	22.3	29.95	1.1mm	350	Aluminium
15	16008	20	34	9	24.7	28.5	0.3cm	130	Pavement

## The Tabulated Text File Format

Here is an example of a tabulated file format. You can use your favorite text editor to create this design table. Use the Tab key to skip from one column to the other. Unit rules are the same as for the Excel sheets.

Designation	d(mm)	D(mm)	B(mm)	d1(mm)	D1(mm)	MinFiletRadius(mm)
623	1.5	5	4	2.6	3.75	0.15
618/4	2	4.5	2.5	2.6	3.75	0.15
624	2	6.5	5	3.35	5.15	0.2
634	2	8	5	4.2	6	0.3
618/5	2.5	5.5	3	3.4	4.6	0.15
61800	5	9.5	5	6.3	8.2	0.3
6000	5	13	8	7.2	10.7	0.3
61802	7.5	12	5	8.95	10.55	0.3
6302	7.5	21	13	11.85	16.95	1
61804	10	16	7	12	14.15	0.3
6404	10	36	19	18.55	27.8	1.1
61806	15	21	7	16.9	19.1	0.3
6306	15	36	19	22.3	29.95	1.1
16008	20	34	9	24.7	28.5	0.3
6308	20	45	23	28.05	37.35	1.5

## The CATIA Design Table

Once it has been read and processed by CATIA, the design table looks something like this:

Design table properties

Name : DesignTable.2  Activity

Comment : This design table was created by CRE on 03/31/99

Configurations Associations

Line	Designation	d	D	B	d1	D1	Material
<1>	623	1.5mm	5mm	4mm	2.6mm	3.75mm	Aluminium
2	618/4	2mm	4.5mm	2.5mm	2.6mm	3.75mm	Iron
3	624	2mm	6.5mm	5mm	3.35mm	5.15mm	Glass
4	634	2mm	8mm	5mm	4.2mm	6mm	Aluminium
5	618/5	2.5mm	5.5mm	3mm	3.4mm	4.6mm	Iron
6	61800	5mm	9.5mm	5mm	6.3mm	8.2mm	Pavement
7	6000	5mm	13mm	8mm	7.2mm	10.7mm	Italian Marble
8	61802	7.5mm	12mm	5mm	8.95mm	10.55mm	Iron
9	6302	7.5mm	21mm	13mm	11.85mm	16.95mm	Pavement
10	61804	10mm	16mm	7mm	12mm	14.15mm	Aluminium
11	6404	10mm	36mm	19mm	18.55mm	27.8mm	Glass
12	61806	15mm	21mm	7mm	16.9mm	19.1mm	Pavement
13	6306	15mm	36mm	19mm	22.3mm	29.95mm	Aluminium

Duplicate data in CATIA model

Buttons: Edit table..., OK, Apply, Cancel

No units

Duplicates the design table external data into the CATIA document. Check this box whenever you intend to re-access your design table on another platform (Unix or Windows NT).

Displays the design table raw data.

Values with units

 About the Design Table

 Using a Design Table

 Programming the Design Table

# Design Table

## Create a design table from the document parameter values



Check the option "Create a design table with current parameter values". Select the parameters to insert, then specify the .xls (Windows™) or .txt file where the design table is to be created.

## Create a design table from a pre-existing file



Check the option "Create a design table from a pre-existing file". Specify the file containing the design table data, then create the necessary associations.

## Edit a design table

In the specification tree, double-click the design table.

or

In the specification tree, right-click the design table to be edited, then select the *DesignTable.object->Edition* function from the contextual menu.

or

If a parameter is constrained by a design table, double-click it in the specification tree, then click the design table icon in the "Edit parameter" dialog box.

## Delete a design table

In the specification tree, right-click the design table to be deleted, then select the Delete function from the contextual menu.

## Program the design table access


- [CellAsReal](#)  
Returns the contents of a cell located in a column intended for real values.
- [CellAsBoolean](#)  
Returns the contents of a cell located in a column intended for boolean values.
- [CellAsString](#)  
Returns the contents of a cell located in a column.
- [CloserInfConfig](#)  
Returns the configuration which contains the largest values less or equal to the values of the given arguments.
- [CloserSupConfig](#)  
Returns the configuration which contains the smallest values greater or equal to the values of the given arguments.
- [CloserValueInfInColumn](#)  
Scans the values of a column and returns the smallest cell value which is the nearest to a specified one.
- [CloserValueSupInColumn](#)  
Scans the values of a column and returns the greatest cell value which is the nearest to a specified one.
- [LocateInColumn](#)  
Returns the index of the first row which contains a specified value.
- [MaxInColumn](#)  
Returns the greatest of a column values.
- [MinInColumn](#)  
Returns the smallest of a column values.
- [Question](#)  
Displays a message in a dialog box, waits for the user to click a button and returns a value indicating which button the user clicked.

# Creating a Design Table with the Current Parameter Values

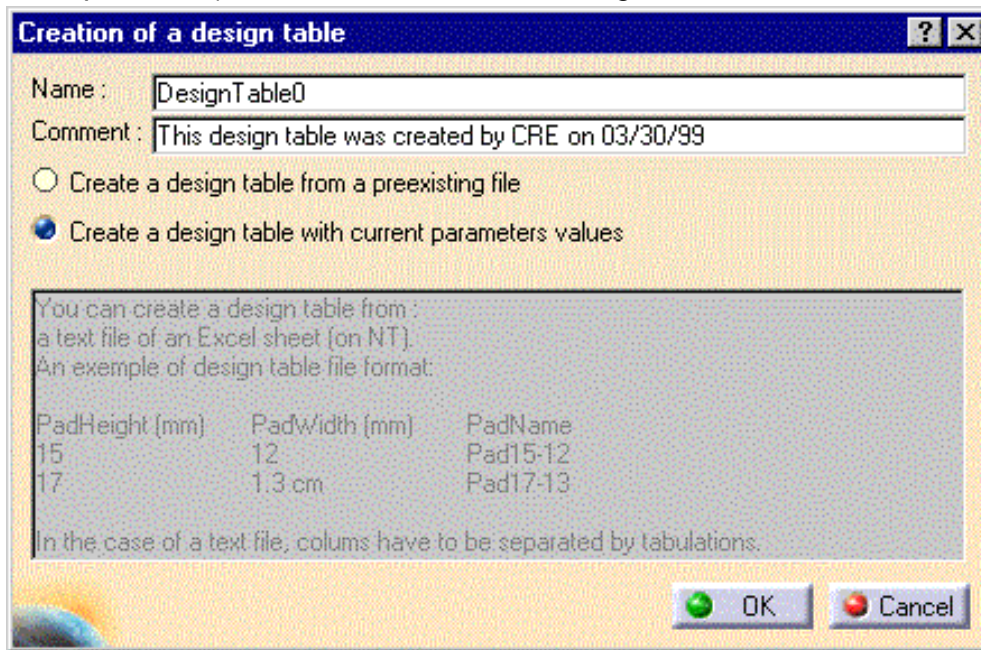
An external design table is created from parameters belonging to the current document. You just have to select the parameters you want to insert in the design table.



1. Open the KwrStartDocument.CATPart document.

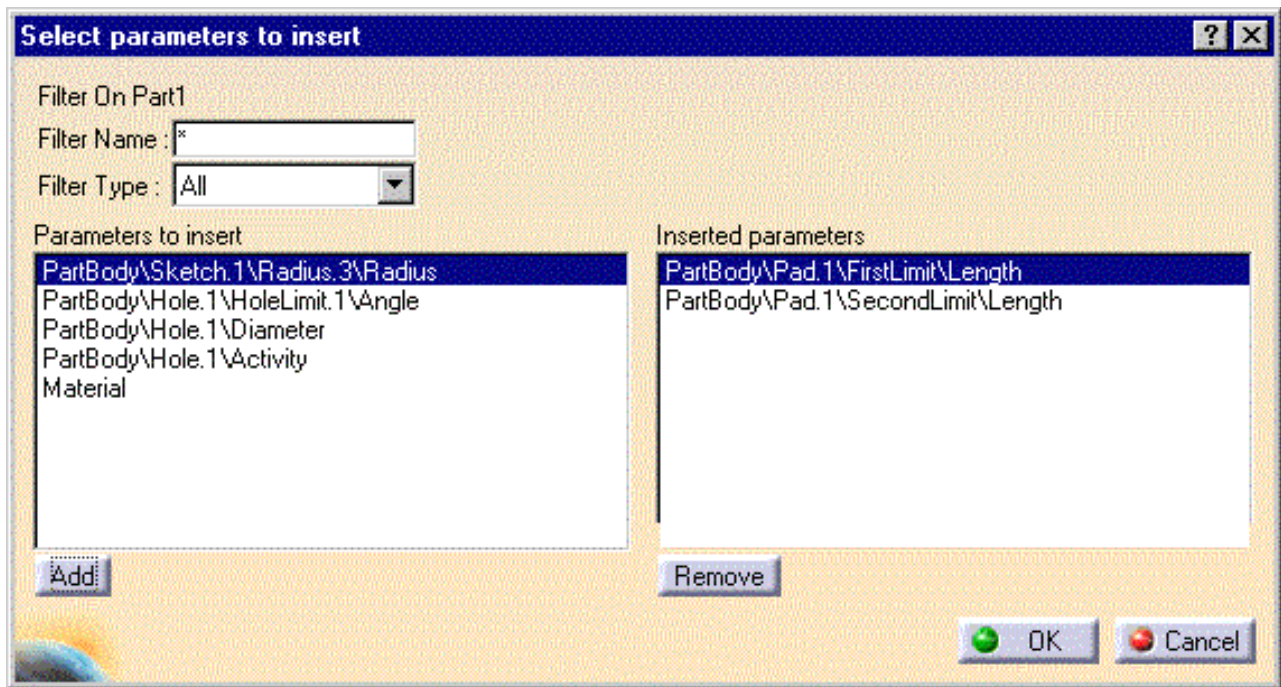
2. Click the  Design Table icon in the standard toolbar.

The "Creation of a Design Table" dialog box is displayed. Enter a name (DesignTable0 in the example below) and a comment for the design table.

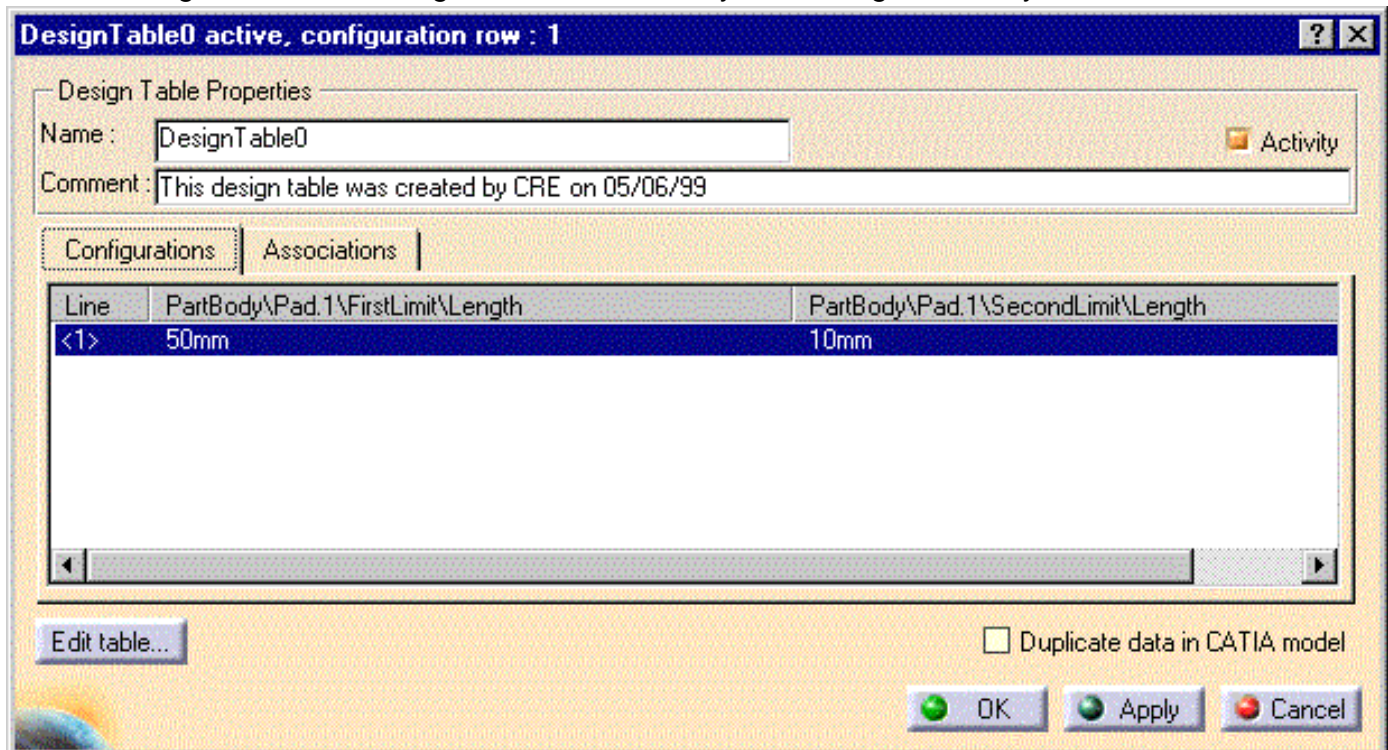


3. Check the 'Create a design table with current parameter values' option. Click OK. The 'Select parameters to insert' dialog box is displayed.

4. In the 'Parameters to insert' list, select the PartBody\Pad.1\FirstLimit\Length then click Add. Repeat the same operation with the PartBody\Pad.1\SecondLimit\Length item. Both items are displayed in the 'Inserted parameters' list.



5. Click OK. A file selection box is displayed.
6. Specify the pathname of the design table to be created. Click OK in the file selection dialog box. The DesignTable0 feature is added to the specification tree and a dialog box displays the newly created design table. This design table contains only one configuration. By default it is active.



7. Click Edit table... to start an Excel application (under Windows NT) or open the text editor under Unix.  
Replace the PartBody\Pad.1\FirstLimit\Length parameter value with 80mm.
8. Save your Excel or .txt file and close your application.
9. Click Apply into the CATIA design table dialog, the document is updated as well as the CATIA design table.
10. Click Cancel if you want to undo your last interaction. Otherwise click OK.  
KwrDesignTable0.CATPart is the resulting document. KwrCreatedDesignTable.xls is the design table created.



A design table can only be created from *non-constrained parameters*, i.e. from parameters which are neither referred to in an active design table nor used in any other *active relation*. If you keep the 'Activity' option checked for DesignTable0 and try to create another design table, you will have to select the parameters to add to your second design table among a restricted parameter list. Uncheck the 'Activity' option if you want to deactivate a design table and reuse its parameters in another design table.



[Up](#)



[Creating a Design Table from Creating a Design Table from](#)




# Creating a Design Table from a Pre-existing File

A design table can be created from a pre-existing file:

- either by associating automatically the document parameters with the table columns. All the parameters common to your document and the input file columns are added to the design table
- or by associating one by one the document parameters with the input file columns.

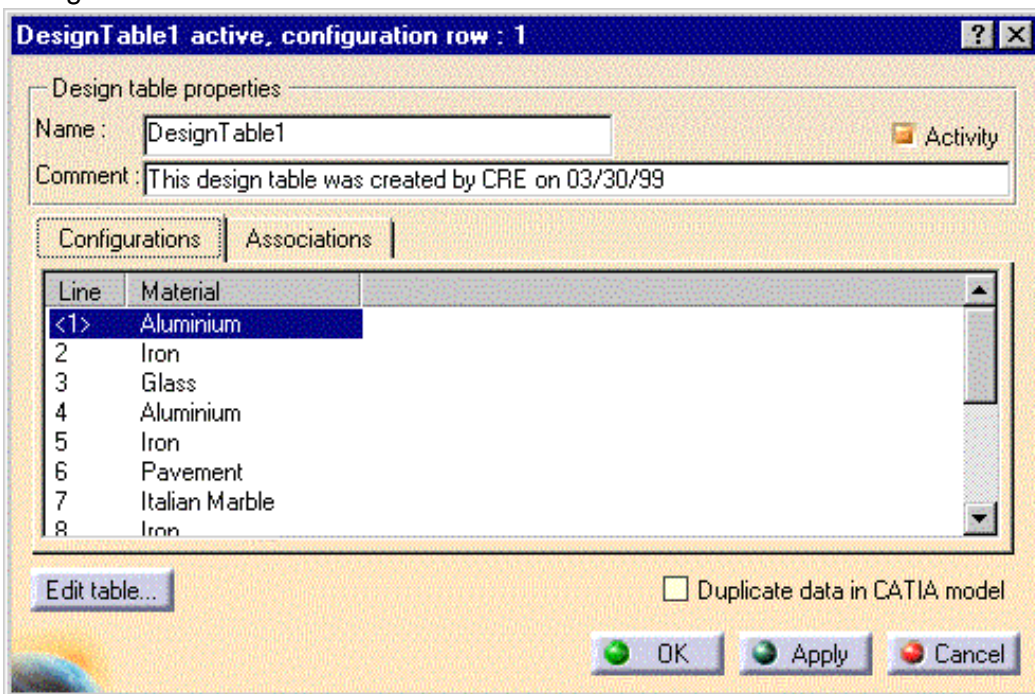
## Associating Automatically Table Columns with Parameters



1. Open the KwrStartDocument.CATPart document.
2. Click the  Design Table icon in the standard toolbar. The "Creation of a Design Table" dialog box is displayed. Enter a name (DesignTable1 for example) and a comment.
3. Check the 'Create a design table from a pre-existing file' option. Click OK.
4. Select the KwrBallBearing.xls file. Click Open. The following dialog box is displayed.



5. Click Yes. The Material parameter is the only one which is common to the document parameters and to the external design table. The following design table is created. The '<' and '>' symbols denote the current configuration.




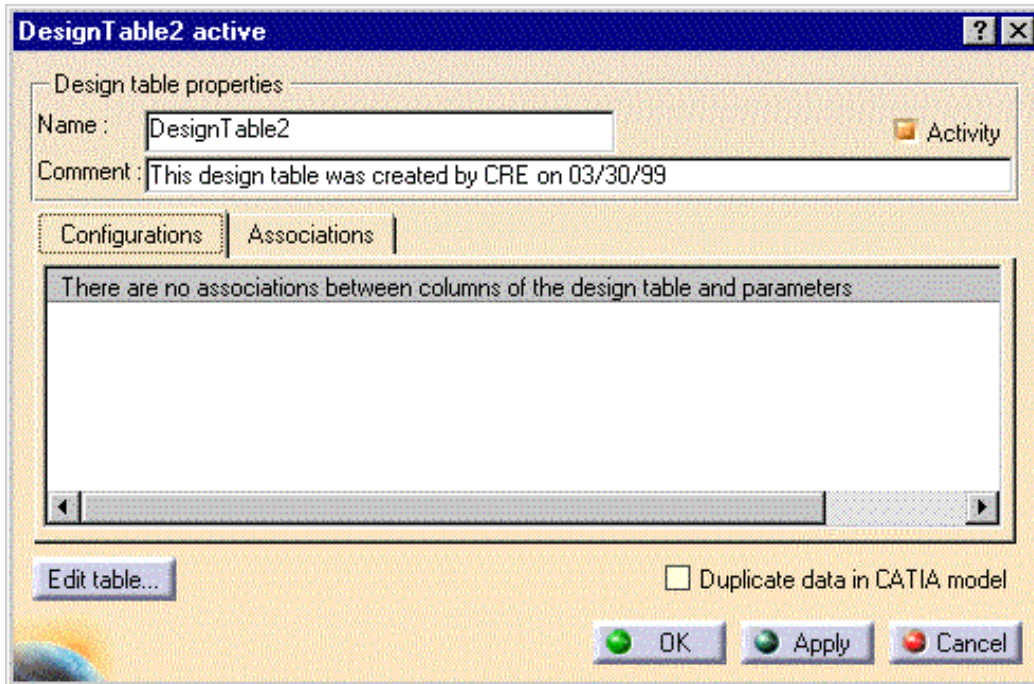
6. Select the configuration you want to apply to the document (line 5 for example). Click Apply. The Iron parameter value is displayed in the specification tree.
7. Click OK to end the design table creation. KwrDesignTable1.CATPart is the resulting document.



## Associating One by One Table Columns with Parameters

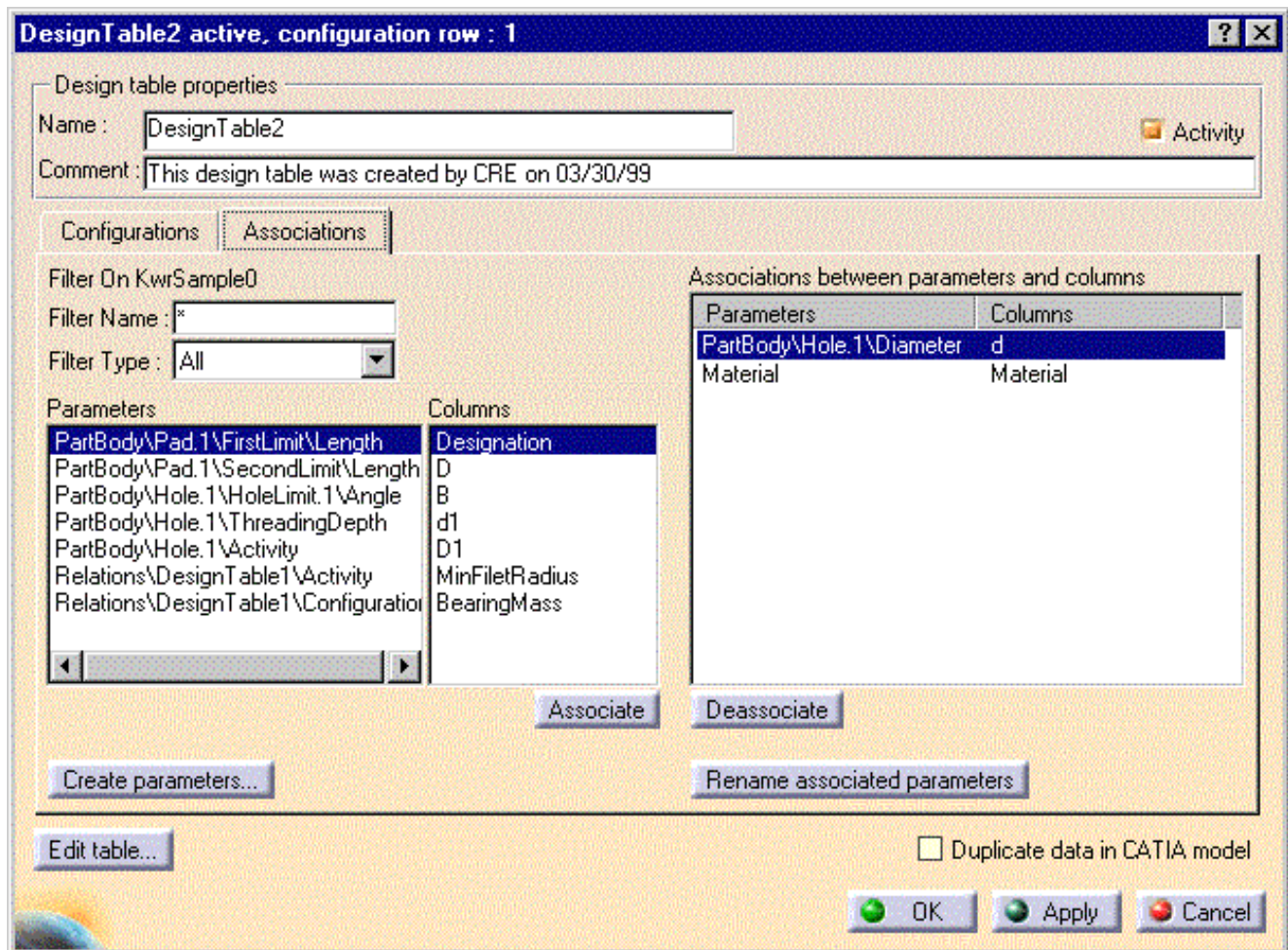


1. Open the KwrStartDocument.CATPart document.
2. Click the  Design Table icon in the standard toolbar. The "Creation of a Design Table" dialog box is displayed. Enter a name (DesignTable2 for example) and a comment.
3. Check the 'Create a design table from a pre-existing file' option. Click OK. A file selection panel is displayed.
4. Select the KwrBallBearing.xls file. Click Open. The "Automatic associations ?" dialog box is displayed.
5. Click No. The following dialog box is displayed.



Now, you have to associate one by one the document parameters with the design table columns.

6. Click the 'Associations' option. The table design dialog box now displays side by side the document parameter list and the input file columns.
7. In the 'Parameters' list, select the `PartBody\Hole.1\Diameter` item. In the 'Columns' list, select the `d` parameter. Then click Associate. A parameter couple is now displayed in the 'Associations between parameters and columns' list.
8. Repeat the same operation for the Material parameter.



Selecting a parameter or an association in the list highlights the corresponding values in the geometry area. The parameter list can be filtered:

- either by clicking on a feature (either in the specification tree or in the geometry area). All the parameter values of the selected feature (and children) are highlighted in the geometry area. The parameter list displays only the parameters of the selected features (and children).
- by specifying a string in the 'Filter Name' field. For example, typing \*ength\* displays all Length parameters
- by specifying a type in the 'Filter Type' field.

The 'Create parameters...' button allows you to create automatically parameters and associations for items of the 'Columns' list. The 'Rename associated parameters' button replaces the parameter name with the column name.

9. Click OK to end the DesignTable2 creation dialog.

The DesignTable2 feature is added as a relation to the specification tree. Double-click DesignTable2 in the specification to edit the table. By default, the configuration <1> is applied to the document. A new material (Aluminum) is applied to the document and the hole diameter is modified. KwrDesignTable2.CATPart is the resulting document.

You can select another configuration and apply it to your document.

As long as a design table is active, the parameters which are declared in it are constrained parameters and you are not allowed to modify them.

Double-clicking a design table in the specification tree displays the design table with its set of configurations and allows you to select a new configuration.



## Programming the Design Table Access

CATIA provides you with functions to access the values of a design table. These functions are described below. You will have to use them with other functions and parameters in formulas, rules or checks (for those of you using the Knowledge Advisor product).

- [CellAsReal](#)
- [CellAsBoolean](#)
- [CellAsString](#)
- [CloserInfConfig](#)
- [CloserSupConfig](#)
- [CloserValueInfInColumn](#)
- [CloserValueSupInColumn](#)
- [LocateInColumn](#)
- [MaxInColumn](#)
- [MinInColumn](#)
- [Question](#)

You don't need to create associations interactively to use these functions.

In the design table function documentation, you can access code samples by clicking 'Sample code'. Click 'Design Table' to display the related design table. When you find an interesting sample, you can Copy/Paste it to your formula, rule or check editor. The KwrProgramDT.CATPart document illustrates how to use the design table access functions.

**Values returned by functions must be declared as user parameters before being used in relations.**



# CloserSupConfig Function

---

## Description

Returns the configuration which contains the smallest values greater or equal to the values of the given arguments. When several configurations meet this condition, the function sorts out the possible configurations with respect to the column order as it is specified in the argument list.

## Syntax

**CloserSupConfig**(*DesignTableName*, *RowName1*, *Value1*, *RowName2*, *Value2* ... )

The **CloserSupConfig** function takes the following arguments:

<i>Argument</i>	<i>Description</i>
<i>DesignTableName</i>	Required. Should be put in quotes.
<i>RowNamei</i>	Should be put in quotes. At least, one couple RowName/Value is required
<i>Valuei</i>	Required. You should specify the units.

## Example

Given the design table below:

	SketchRadius (mm)	PadLim1 (mm)	PadLim2 (mm)
1	120	60	10
2	130	50	30
3	120	60	25
4	140	50	40

The statement below returns 3:

```
CloserSupConfig("DT1", "PadLim1", 60mm, "SketchRadius", 120mm, "PadLim2", 20mm)
```

---

# Question Function

---

## Description

Displays a message in a dialog box, waits for the user to click a button and returns a value indicating which button the user clicked (true if Yes was clicked, false if No was clicked)

## Syntax

**Question**(*String* [# *String1* # *String2* ..., *Param1Name*, *Param2Name*, ...] )

The **Question** function takes one required argument and several optional arguments depending on whether parameter values are to be displayed in the message.

<i>Argument</i>	<i>Description</i>
<i>String</i>	Required. String to be displayed in the dialog box (should be put in quotes).
# <i>String1</i> , <i>Param1Name</i> ...	Optional. When parameter values are to be displayed within the message, the arguments should be specified as follows: <ul style="list-style-type: none"><li>● one string in quotes including a # symbol wherever a parameter value is to be displayed</li><li>● as many [, <i>parameter name</i>] statements as parameter values declared with a "#" in the message.</li></ul>

Use the "|" symbol to insert a carriage return in a prompt.

## Example

```
Boolean2 =  
Question("SketchRadius is # | Do you want to change this value ?",PartBody\Sketch.1\Radius.3\Radius )
```

---

# Using a Design Table

A design table is a feature that you create either from your document parameters or from external data. No matter the existence of external data, you must **create** the design table in CATIA. The design table creation process includes the following steps:

## [Creating a design table from a pre-existing file](#)

1. select the pre-existing file containing the raw data.
2. create the associations between the document parameters and the external table columns. You can choose to create these associations automatically.
3. edit the CATIA design table generated.
4. select a configuration in the generated design table. You can modify the default configuration proposed by CATIA.
5. apply the design table feature to your document.

or

## [Creating a design table from current parameter values](#)

1. create a table from the document parameters.
2. select the parameters to add to the design table.
3. specify a file to contain the design table generated.
4. edit the CATIA design table generated.
5. apply the design table to your document.



A design table is a knowledgware feature which behaves like any CATIA feature. For example, you can edit it from the specification tree by double-clicking it or by selecting the *feature.object*->Definition function from the contextual menu.


When a parameter is constrained by a design table, the "Edit Parameter" dialog box which is displayed when you double-click it in the specification tree, gives you an access to the design table through the design table icon.



# Behavior

## Create a behavior



Click the  icon and write your script in the editor which is displayed.

## Manipulate automation objects



You can only manipulate the objects which are below the [Shape](#) object in the automation object hierarchy.

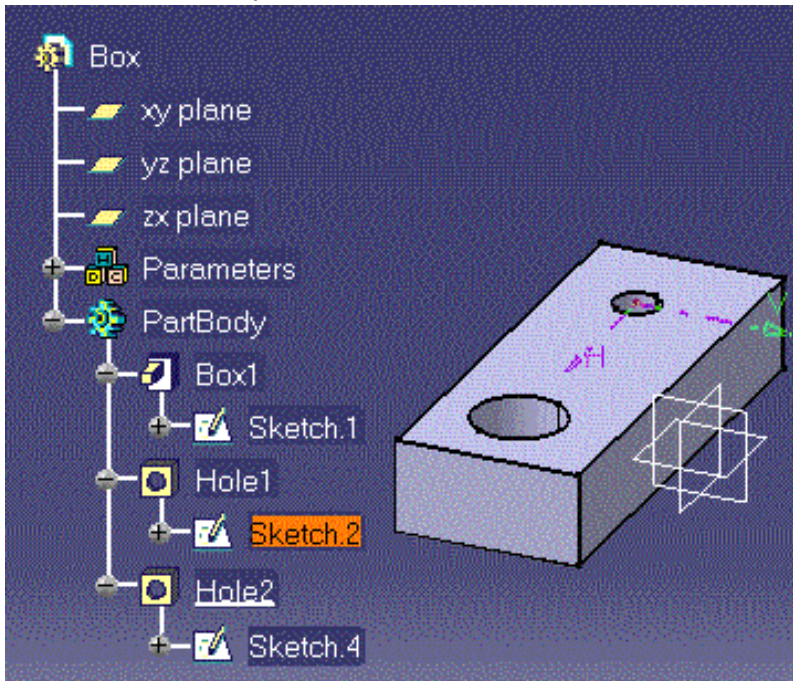
# Creating a Behavior



A behavior is a set of VB Script operations you associate with a feature. These operations are executed when the feature responds to a given event. At present, only a feature which is dragged and dropped can be assigned a behavior. This task explains how to proceed to create a behavior.



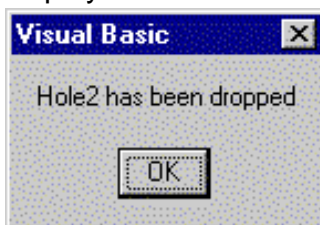
1. Create a box or a pad similar to the one below:



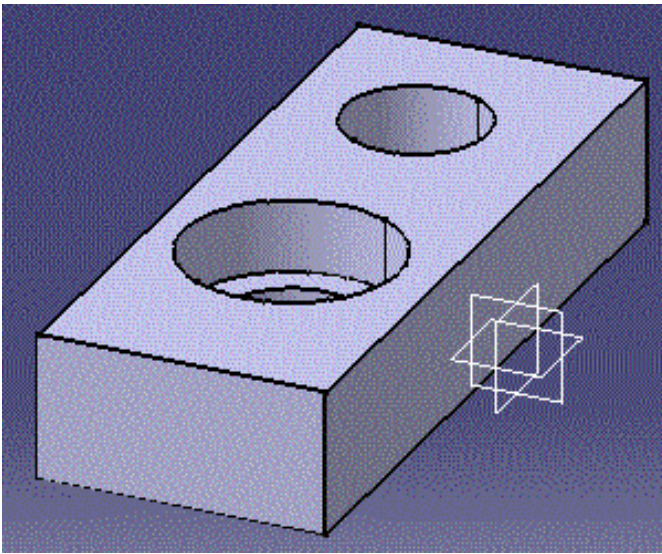
2. Select the Box item in the specification tree.
3. In the Start menu, select Infrastructure-> Knowledge Advisor workbench
4. Select the Hole2 feature either from the geometry area or from the specification tree
5. Click the Behavior icon. The behavior dialog box is displayed. At present the list of events that can be referred to in a behavior is limited to the Drag & Drop event.
6. Enter a behavior name
7. Copy/Paste the code below from your browser to the edition box of the behavior editor.

```
MsgBox("Hole2 has been dropped")  
MyEvent.GetDroppedFeature.Type = 2
```

8. Click OK. A behavior is added to the specification tree.
9. Go back to the Part Design workbench (double-click the document in the geometry area for example)
10. Drag and drop the Hole2 feature so that it gets closer to Hole1. The message box below is displayed:



This is what you can see now onscreen. The Hole2 feature has been converted to a type2 hole (i.e. a counterbored hole).



This example is a simple illustration of how you can create a behavior and how it operates when it is applied to a feature. To create a full-featured behavior application, you must have a good handle on the VB Script language as well as all the automation objects that can be accessed in VB Script.



[Up](#)



[Creating a Behavior](#)



[Manipulating Feature Attribut](#)

# Manipulating Feature Attributes




Not all the features can be accessed and modified in a behavior. You can only manipulate the objects which are below the [Shape](#) object in the automation object hierarchy. Here is an example which illustrates how to manipulate feature attributes.



The feature to be dragged and dropped is a simple type hole. When the behavior is executed:

1. the following hole properties are displayed in a Visual Basic message box:
  - the feature type,
  - the feature name
  - the hole diameter.
2. the user is prompted
  - either to click OK to modify the hole properties after it is dropped. New property values will be:
    - hole (type = 2
    - diameter = 10.0
    - head diameter = 20.0.
  - or to click Cancel and just drop the initial hole without modifying its properties.

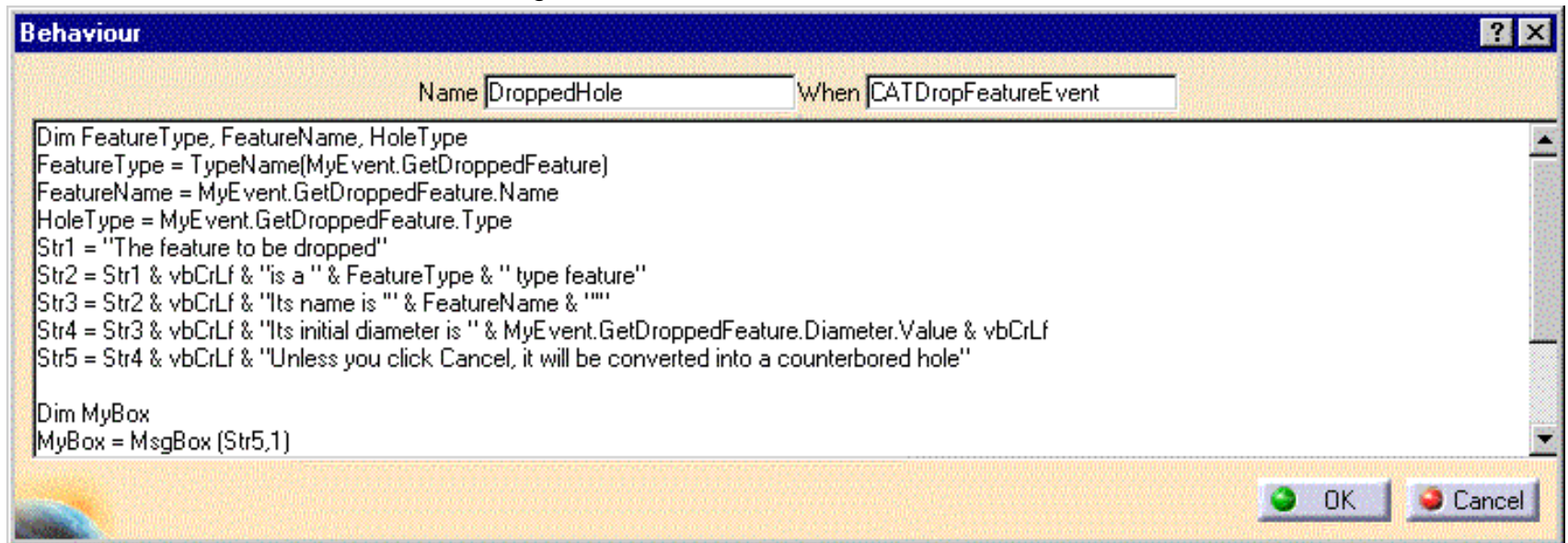


1. Create a pad with at least a simple type hole to be dragged and dropped.
2. Select the root feature and access the Knowledge Advisor workbench.
3. Select any hole feature either in the geometry area or in the specification tree
4. Click the  icon. The Behavior editor is displayed.
5. Copy/Paste the code below from your browser to the behavior editor.

```
Dim FeatureType, FeatureName, HoleType
FeatureType = TypeName(MyEvent.GetDroppedFeature)
FeatureName = MyEvent.GetDroppedFeature.Name
HoleType = MyEvent.GetDroppedFeature.Type
Str1="The feature to be dropped"
Str2=Str1 & vbCrLf & "is a " & FeatureType & " type feature"
Str3=Str2 & vbCrLf & "Its name is '" & FeatureName & "'"
Str4=Str3 & vbCrLf & "Its initial diameter is " & MyEvent.GetDroppedFeature.Diameter.Value & vbCrLf
Str5=Str4 & vbCrLf & "Unless you click Cancel, it will be converted into a counterbored hole"
```

```
Dim MyBox
MyBox = MsgBox (Str5,1)
if MyBox = 1 then
MyEvent.GetDroppedFeature.Type = 2
MyEvent.GetDroppedFeature.HeadDiameter.Value = 20.0
MyEvent.GetDroppedFeature.Diameter.Value = 10.0
End If
```

The behavior editor looks now something like this:



6. Click OK. The DroppedHole behavior is added to the Hole feature in the specification tree.
7. Go back to the Part Design workbench (double-click the document for example).
8. Drag and drop the Hole object. The Hole attribute values are displayed within a Visual Basic dialog box.
9. Click OK to convert the dragged hole into a counterbored hole or click Cancel to drop it and don't modify its properties.



The Name property returns the name of the feature after it has been dragged. This name is not the initial feature name. It is the name that CATIA assigns by default to the dropped feature.



Up



Creating a Behavior



Manipulating Feature Attributes

# Workbench Description

## The Knowledge Advisor Menu Bar

The menu bar which is available in the Expert Knowledge workbench is the standard one. Refer to the [CATIA Menu Bar](#) for more information.

## The Knowledge Advisor Toolbar

The figure below describes the Knowledge Advisor toolbar.



On the figure opposite, click an icon to display the documentation of the task directly associated with the icon.

Here is a brief description of each icon.



The **Rule Editor** icon provides access to the rule editor. Click this icon to create a rule, write its code, test its syntax and apply it to your document.



The **Check Editor** icon provides access to the check editor. Click this icon to create a check, write its code, tests its syntax and apply it to your document.



The **Knowledge Inspector** icon allows you to query a design to determine and preview the results of changing any parameters without committing themselves to actually changing the design.



The **Behavior** icon is a means to create a script specifying how to change some feature attributes when an event occurs. Clicking this icon opens a behavior editor. A behavior is to be written in Visual Basic language.

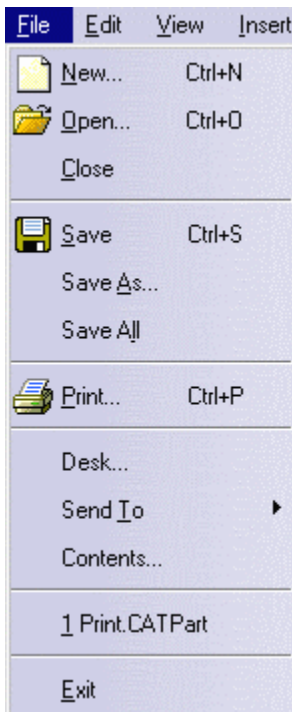
This section presents the main menu bar available when you run the application and before creating or opening a document.



## Start

The Start menu is a navigation tool intended to help you toggle between different workshops. The contents of the Start menu vary according to the configurations and/or products installed. For more information about the Start menu, refer to "[Accessing the Navigation Tools](#)".

## File

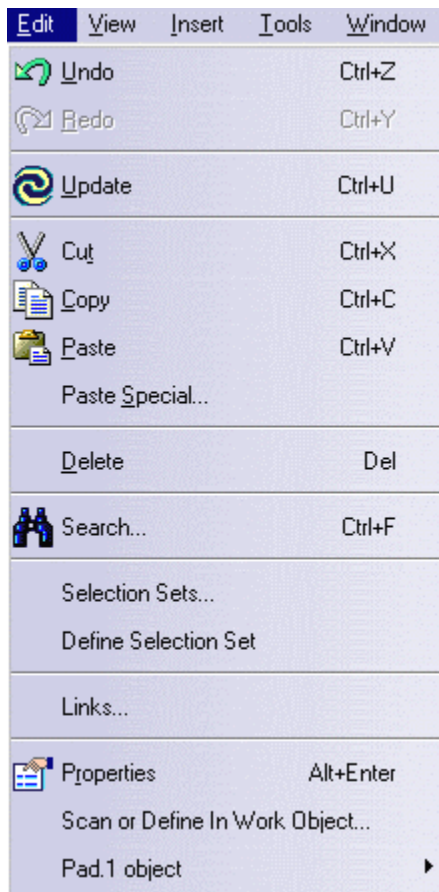


For...	See...
New...	<a href="#">Creating New Documents</a>
Open...	<a href="#">Opening Existing Documents</a>
Close	<a href="#">Closing Documents</a>
Save	<a href="#">Saving Existing Documents</a>
Save As	<a href="#">Saving Documents For the First Time or Under Another Name</a> <a href="#">Saving Documents In Other Formats</a>
Save All	<a href="#">Saving All Documents</a>
Print...	<a href="#">Customizing Print Settings Before Printing Your Documents</a>
Desk...	<a href="#">Using the FileDesk Workbench</a>
Send To	<a href="#">Transferring CATIA Version 5 Data</a>

## Edit



For...	See...
Undo	<a href="#">Undoing Actions</a>
Redo	<a href="#">Recovering Last Action Undone</a>



Cut	<a href="#">Cutting and Pasting Objects</a>
Copy	<a href="#">Copying and Pasting Objects</a>
Paste	<a href="#">Cutting and Pasting Objects</a> <a href="#">Copying and Pasting Objects</a>
Delete	<a href="#">Deleting Objects</a>
Search...	<a href="#">Selecting Using the Search... Command</a>
Selection Sets...	<a href="#">Storing Selections Using Selection Sets</a>
Define Selection Set	<a href="#">Storing Selections Using Selection Sets</a>
Links...	<a href="#">Editing Document Links</a>
Properties...	<a href="#">Displaying and Editing Graphic Properties</a>

## View



<b>For...</b>	<b>See...</b>
Toolbars	<a href="#">Viewing and Hiding Toolbars</a>
Geometry	<a href="#">Setting Document Window Layout Preferences</a>
Specifications	<a href="#">Setting Document Window Layout Preferences</a>
Compass	<a href="#">About the 3D Compass</a>
Reset Compass	<a href="#">Manipulating Objects Using the Mouse and Compass</a>
Overview	<a href="#">Using the Overview on the Specification Tree</a>
Fit All In	<a href="#">Fitting All Geometry in the Geometry Area</a>
Zoom Area	<a href="#">Zooming In On An Area</a>
Zoom In Out	<a href="#">Zooming In</a> <a href="#">Zooming Out</a>
Pan	<a href="#">Panning</a>
Rotate	<a href="#">Rotating</a>
Modify->Look At	<a href="#">Looking At Objects</a>



Modify->Turn Head, Zoom In,  
Zoom Out, Normal View

[Turning Your Head To View An Object](#)  
[Zooming In](#)  
[Zooming Out](#)  
[Viewing Along a Normal to a Plane](#)

Named Views...

[Using Standard and User-Defined Views](#)

Render Style

[Using Rendering Styles](#)

Navigation Mode

[Navigating](#)

Lighting...

[Setting Lighting Effects](#)

Depth Effect...

[Setting Depth Effects](#)

Ground

[Viewing Objects against the Ground](#)

Magnifier...

[Magnifying](#)

Hide/Show

[Hiding and Showing Objects](#)

Full Screen

[Using the Full Screen](#)

## Tools



**For...**

**See...**

Formula...

[Using Knowledgeware Capabilities](#)

Image

[Capturing and Managing Images for the Album](#)

Macro

[Recording, Running and Editing Macros](#)

Customize...

[Customizing Toolbars](#)

Options...

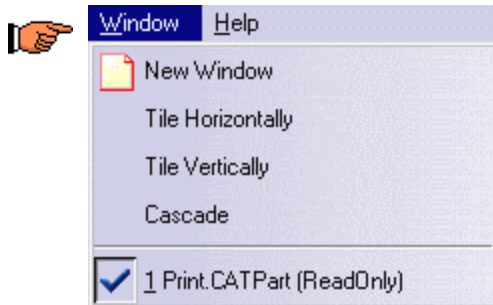
[Customizing Settings](#)



Search Order...

[Creating a Document Search Order](#)

## Window



**For...**

**See...**

New Window

[Using Document Windows](#)

Tile Horizontally

[Using Document Windows](#)

Tile Vertically

[Using Document Windows](#)

Cascade

[Using Document Windows](#)

## Help



**For...**

**See...**

CATIA V5.2 Help

[Getting Contextual Help](#)

Contents, Index and Search

[Accessing the Online Help Library](#)

What's This?

[Using the What's This? Command](#)

User Galaxy

[Accessing the Dassault Systèmes User Galaxy](#)

About CATIA V5.2

[Displaying Copyright Information](#)

# Quick Reference

## Parameters

### Create a user parameter



Select a type in the 'New Parameter of type' list, then click 'New Parameter of type'.

### Edit/modify a user parameter



Select the parameter to be edited in the "Formulas" dialog box, then modify its value in 'Edit name, value or formula' .

or

In the specification tree, double-click the parameter to be edited, then modify its value in the "Edit Parameter" editor.

or

In the specification tree, right-click the parameter to be edited, then select the *Parameter.object->Definition* function from the contextual menu.

### Add a comment to a user parameter



In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula', then select 'Edit Comment...' from the contextual menu.

or

Whatever the edition box, right-click the value field(s), then select 'Edit comment...' from the contextual menu.

### Add a tolerance to a magnitude type parameter



In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula', then select 'Add Tolerance' from the contextual menu.

or

Whatever the edition box, right-click the value field(s), then select 'Add Tolerance...' from the contextual menu.

### Specify lower and upper bounds



In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula', then select 'Add Range' from the contextual menu.

Whatever the edition box, right-click the value field(s), then select 'Add Range...' from the contextual menu.

### Specify an increment/decrement amount




In the "Formulas" dialog box, right-click the value field in 'Edit name, value or formula', then select 'Change Step' from the contextual menu.

or

Whatever the edition box, right-click the value field(s), then select 'Change Step...' from the contextual menu.

### Specify the Material parameter value

Select the Material parameter in the specification tree, then click .

or

Double-click the Material parameter in the specification tree, then enter the material value in the dialog box.

or



In the "Formulas" dialog box, select the Material parameter in the parameter list, then modify its value in 'Edit name, value or formula'.

### Update the Mechanical\_Property parameters

Select the Properties->Mass option from the feature contextual menu and click OK.

### Import parameters



In the "Formulas" dialog box, click `Import . . .`. A file selection dialog box is displayed. Select either a .xls file(Windows NT only) or a .txt file.

### Delete a parameter



Select the parameter to be deleted in the "Formulas" dialog box, then click 'Delete Parameter'.

or In the specification tree, right-click the parameter to be deleted, then select the Delete function from the contextual menu.

# Formulas

## Create a formula



In the "Formulas" dialog box, select the parameter to be constrained, then click 'Add Formula'. Enter the formula in the "Formula Editor".

or



In the "Formulas" dialog box, select the parameter to be constrained, then enter the formula in 'Edit name, value or formula'.

or



In the "Formulas" dialog box, double-click the parameter to be constrained and enter the formula in the "Formula Editor".

or

When editing a parameter, right-click the value field(s), then select Formula->Edit from the contextual menu.

## Edit/modify a formula



In the parameter list of the "Formulas" dialog box, select the parameter defined by the formula, then modify the formula in 'Edit name, value or formula'.

or

In the specification tree, right-click the formula to be edited, then select the Formula.object->Definition function from the contextual menu.

or

In the specification tree, double-click the formula to be edited.

or

In the specification tree, double-click the user parameter, then click the  $f(x)$  button in the "Edit Parameter" dialog box.

or

When editing a parameter, right-click the value field(s), then select Formula->Edit from the contextual menu.

## Activate/deactivate a formula



In the parameter list of the "Formulas" dialog box, select the parameter which is constrained by the formula to (de)activate and use the Activate/Deactivate check box right of the 'Edit name, value or formula' field.

or



In the parameter list of the "Formulas" dialog box, select the formula/activity parameter and modify its value in 'Edit name, value or formula'.

or

In the specification tree, right-click the formula whose activity is to be modified, then select the *Formula.object*->(De)activate function from the contextual menu.

or

When editing a parameter constrained by a formula, right-click the value field(s), then select the Formula->(De)activate function from the contextual menu.

## Import parameters and formulas



In the "Formulas" dialog box, click 'Import', then specify the import file path. The import file should be either a .txt file or a .xls file.

## Delete a formula



In the parameter list of the "Formulas" dialog box, select the parameter which is constrained by the formula to be deleted, then click 'Delete Formula'.

or

In the specification tree, right-click the formula to be deleted, then select the Delete function from the contextual menu.

or

When editing a parameter constrained by a formula, right-click the value field(s), then select the Formula->Delete function from the contextual menu.

# Rules

## Create a rule



Enter the rule name and comments. Then write your rule in the rule editor. To enter a parameter within a rule, you can:

- either click the parameter in the specification tree
- or click the dimension displayed in the geometry area
- or use the editor dictionary.

### Edit/modify a rule

In the specification tree, double-click(twice) the rule to be edited.

or

In the specification tree, right-click the rule to be edited, then select the *Rule.object*->Edition function from the contextual menu.

### Activate/deactivate a rule

In the specification tree, right-click the rule to be activated or deactivated, then select the *Rule.object*->(De)activate function from the contextual menu.

or



In the parameter list of the "Formulas" dialog box, select the rule/activity parameter modify its value in 'Edit name, value or formula'.

### Delete a rule

In the specification tree, right-click the rule, then select the Delete function from the contextual menu.

## Checks

### Create a check



Enter the check name and comments. Then write your check in the check editor. To enter a parameter within a check, you can:

- either click the parameter in the specification tree
- or click the dimension displayed in the geometry area
- or use the Dictionary.

## Edit/modify a check

In the specification tree, double-click the check to be edited.

or

In the specification tree, right-click the check to be modified, then select the *Check.object->Edition* function from the contextual menu.

## Activate/deactivate a check

In the specification tree, select the check to be activated or deactivated, then select the *Check.object->(De)activate* function from the contextual menu.

or



In the parameter list of the "Formulas" dialog box, select the check/activity parameter and modify its value in 'Edit name, value or formula'.

## Delete a check

In the specification tree, right-click the check to be deleted, then select the Delete function from the contextual menu.

# Design Table

## Create a design table from the document parameter values



Check the option "Create a design table with current parameter values". Select the parameters to insert, then specify the .xls (Windows™) or .txt file where the design table is to be created.

## Create a design table from a pre-existing file



Check the option "Create a design table from a pre-existing file". Specify the file containing the design table data, then create the necessary associations.

## Edit a design table

In the specification tree, double-click the design table.

or

In the specification tree, right-click the design table to be edited, then select the *DesignTable.object->Edition* function from the contextual menu.

or

If a parameter is constrained by a design table, double-click it in the specification tree, then click the design table icon in the "Edit parameter" dialog box.

## Delete a design table

In the specification tree, right-click the design table to be deleted, then select the Delete function from the contextual menu.

## Program the design table access


- [CellAsReal](#)  
Returns the contents of a cell located in a column intended for real values.
- [CellAsBoolean](#)  
Returns the contents of a cell located in a column intended for boolean values.
- [CellAsString](#)  
Returns the contents of a cell located in a column.
- [CloserInfConfig](#)  
Returns the configuration which contains the largest values less or equal to the values of the given arguments.
- [CloserSupConfig](#)  
Returns the configuration which contains the smallest values greater or equal to the values of the given arguments.
- [CloserValueInflnColumn](#)  
Scans the values of a column and returns the smallest cell value which is the nearest to a specified one.
- [CloserValueSuplnColumn](#)  
Scans the values of a column and returns the greatest cell value which is the nearest to a specified one.

- [LocateInColumn](#)  
Returns the index of the first row which contains a specified value.
- [MaxInColumn](#)  
Returns the greatest of a column values.
- [MinInColumn](#)  
Returns the smallest of a column values.
- [Question](#)  
Displays a message in a dialog box, waits for the user to click a button and returns a value indicating which button the user clicked.

## Behavior

### [Create a behavior](#)



Click the  icon and write your script in the editor which is displayed.

### [Manipulate automation objects](#)



All objects below the Shape object can be manipulated in a behavior.

# Case Study

A bearing is defined by parameters such as its principal dimensions, its basic load ratings, its limiting speeds and its mass. It belongs to a category which corresponds a certain range of its parameter values. In a catalogue, a bearing is referred to by a designation. Bearing types are described by tables which define the bearing parameter values including the designation.

The bearing example has been chosen here because the bearing tables given in distributor and retailer catalogues illustrate quite well the design table principles. The bearing itself is a good example of how components within a mechanical part can be constrained by relations.

## The Data you Need to Perform the Scenario

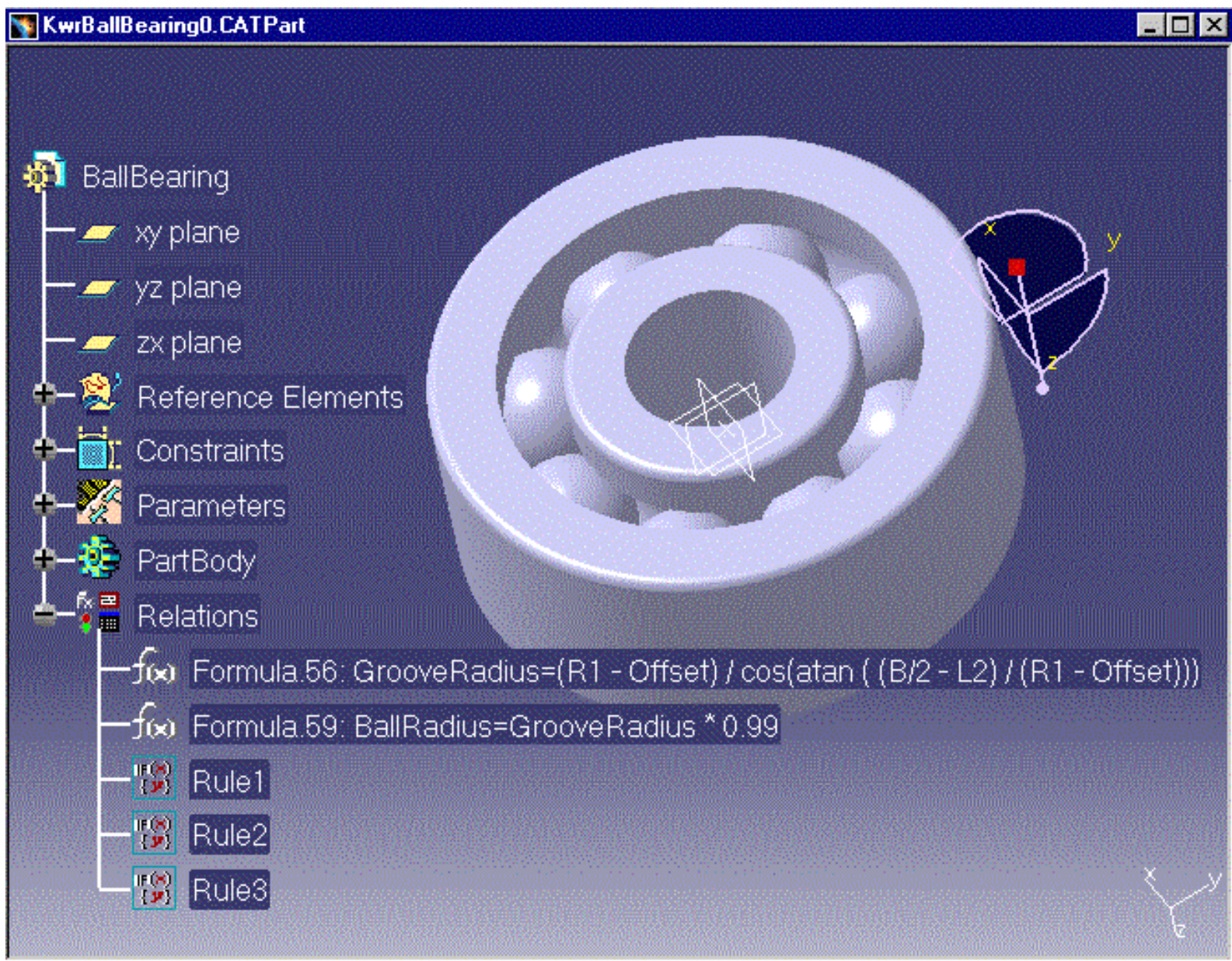
The required data is delivered with the Knowledge Advisor product.

The starting document	KwrBallBearing0. <b>CATPart</b>
The bearing design table	KwrBallBearing.xls
An import file	KwrBallBearingImport.txt
A CATScript macro	KwrMacro1. <b>CATScript</b>

When creating **Rule4** in your own environment, you should replace the pathname given as the argument of the `LaunchMacroFromFile` function with the pathname corresponding to the file where the macro has been unloaded.

## Some Explanations about your Starting Document

The document we start from in this case study is a part **inspired** by the SKF deep groove ball bearing with Designation 623. Our purpose here is not to design an industrial ball bearing but to use a ball bearing as a basis to illustrate the CATIA knowledgware capabilities.



Here are some details about the part design, the document initial parameters and its relations.

## The Document Parameters

The bearing rings are coaxial shafts created from the Sketch.1 and Sketch.2 features. The balls are shafts created from the Sketch.4 feature. To find out the role played by the various parameters, proceed as follows:

1. In the specification tree, expand the PartBody feature and slide the cursor on the Sketch.1 feature. The sketch used to create the outer bearing ring is highlighted in the geometry area.
2. To identify the sketch.1 parameters:
  1. double-click sketch.1 in the specification tree
  2. click the formula icon
  3. in the parameter list:
    - select B.  
The B dimension is highlighted in the geometry area. Its value is set to 4mm. It defines the total bearing width.

- select L2  
The L2 dimension is highlighted in the geometry area. Its initial value is set to 1mm and you can note it is constrained by a formula. Repeat the same operation with L3. L2 and L3 play a symmetric role in the sketch definition.
- select D  
The D dimension is highlighted in the geometry area. Its value is set to 5 mm. D defines the bearing outside **radius** (or external radius of the outer ring) . Repeat the same operation with D1 which defines the internal **radius** of the outer ring.
- CentreAxe1 defines the groove center.

3. Use the same method for the other parameters.

## The Relations

Take a look at the formulas and rules added to the document. The ball radius is constrained by the groove radius which is itself calculated from Sketch.1 parameters. The rules allow you to:

- define geometric parameters with respect to others
- ensure the sketch.1 and sketch.2 consistency
- and define the ball number.


All these relations define the bearing geometry. At this stage of the exercise, it is not recommended to modify them.

## The Step-by-Step Procedure



A design table is created from a pre-existing file. The data set contained in this pre-existing file is quite similar to the data set which identifies a bearing in a catalogue. The design table which is created defines a number of configurations. Applying a new configuration results in a bearing modification.





1. Open the KwrBallBearing0.CATPart document.
2. Click the  Design Table icon in the standard toolbar.
3. Check the "Create a design table from a pre-existing file" option. Click OK.
4. Select the KwrBallBearing.xls file and associate automatically the design table columns and the document parameters.
5. In the "Design table" dialog box, select the configuration 8.  
Your ball bearing has changed. It is now a silver bearing. You can tell the difference when you look at the geometry area. The bearing width is also modified. Click OK to exit the Design Table dialog box. **Keep your document open and proceed to the next task.**



A Cost parameter is defined as a function of the ball number. Creating a check provides you with a means to be warned whenever the bearing cost is too high.





1. Click the  icon.
2. Create a new user parameter of real type and constrained by the  $Cost = 3 * BallNumber$  formula. To do so:
  1. Select Real in the type list and click 'New Parameter of type'.
  2. In 'Edit name, value or formula', replace the Real.1 string with Cost, then press Enter.
  3. Click 'Add Formula'.
  4. Enter the  $3 * BallNumber$  formula in the "Formula Editor". Click OK.
  5. Click OK in the "Formulas" dialog box to add the Cost parameter.
3. Create a check warning you when the Cost value is greater than 80. To do so:
  1. Access the Knowledge Advisor workbench
  2. Click the  icon.
  3. Enter the CostCheck string in the Name field of the first dialog box. Click OK
  4. In the "Check Editor", select the Warning type and enter the string "It is too expensive" in the message field. Then enter the  $Cost \leq 80$  relation in the edition box.
  5. Click OK to create your check and exit the editor. At this stage, no particular message is displayed.
4. In the specification tree, double-click the design table and select the configuration 12. Click OK. The message "It is too expensive" is displayed.

**Keep your document open and proceed to the next task**



A multiple value parameter is created. Depending on this parameter value, a rule displays a message or launches a macro.





1. Click the  icon.
2. Create a new user parameter of string type and multiple values. To do so:
  - Select 'String' in the type list, select 'Multiple values' **then** click 'New Parameter of type'.
  - In the "Value List of String" dialog box, enter one-by-one the step1, step2 and step3 values. Click OK.
  - In 'Edit name, value or formula', replace the String.1 string with Status, then click OK.
3. Create a rule which, depending on the Status parameter value, either displays a message asking you to import a file, or launches a macro, or does not modify anything. To do so:
  - Access the Knowledge Advisor workbench
  - Click the  icon.
  - Enter the **Rule4** string in the Name field of the first dialog box. Click OK

- Copy/Paste the code below into the rule edition box.

```
if Status == "step2"  
Message("Import the KwrBallBearingImport text file")  
else if Status == "step3"  
LaunchMacroFromFile("e:/tmp/KwrMacro1.CATScript")
```

- Click OK to add the rule to the document and execute it.

4. Click the  icon. In the "Formulas" dialog box, select the Status parameter and replace its step1 value with step2. Click OK. A message asks you to import the KwrBallBearingImport text file.
5. Click 'Import' and select the KwrBallBearingImport.txt file. Three parameters are then added to the document. Click OK in the dialog box displaying the parameters and formulas to be imported.
6. Select the Status parameter and replace the step2 value with step3. Click OK. The KwrMacro1.CATScript is executed and a rod is created.

In the final document, double-click the Material parameter in the specification tree. In the "Edit Parameter" box which is displayed, click  to display the design table which constrains the Material parameter.

The resulting document is KwrBallBearingResult.CATPart.



## Glossary

Many of the definitions included in this glossary are only pertinent within the CATIA knowledgeware context.

# Nonalphabetic Terms

## | (operator)

Breaks a single line message into a multiple line message. Can only be used in the Message function when programming rules and checks.

## .CATScript

The extension of a macro file generated by CATIA Version 5. A macro file can be specified as the argument of the LaunchMacroFromFile function which can be called in rules and checks.

## .txt

The extension of a human-readable file composed of text characters. This file format can be used as an import file format when importing parameters and formulas.

## .xls

The extension of an Excel file. This file format can be used as an import file format when importing parameters and formulas under Windows™.

# A

## activity

A property which defines whether a relation is applied to a document or not. The activity value is either `true` or `false`. It is indicated by an icon in the specification tree and can also be read in the document parameter list.

## association

A link between a document parameter and its equivalent parameter in an external design table. Associations are to be created when the document parameter names do not correspond exactly to the parameter names read in the design table.

# B

## behavior

A set of operations to be executed as a reply to a particular event occurring on a feature.

## C

### **check**

A set of statements intended to give you a clue as to whether certain conditions are fulfilled or not. A check does not modify the document it is applied to. A check is a feature. In the document specification tree, it is displayed as a relation that can be activated and deactivated. Like any feature, a check can be manipulated from its contextual menu.

### **configuration**

A row in the design table. A configuration is a consistent set of parameter values that can be applied to a document.

## D

### **design table**

A table containing values to be potentially applied to a document to manage its parameter values. It can be created from the document parameters or from an external file. A design table is a feature. In the document specification tree, it is displayed as a relation that can be activated or deactivated. Like any feature, a design table can be manipulated from its contextual menu.

### **dictionary**

The set of parameters, operators, keywords, functions and other components that make up the language to be used to write formulas, rules and checks. The formula, rule and check editors provide you with an interactive view of the dictionary.

## F

### **formula**

A relation specifying a constraint on a parameter. The formula relation is a one-line statement. Its left part is the parameter to be constrained, the right part is a relation taking as its variables other parameters. A formula is a feature. In the document specification tree, it is displayed as a relation that can be activated or deactivated. Like any feature, a formula can be manipulated from its contextual menu.

# K

## knowledgeware

The set of software components dedicated to the creation and manipulation of knowledge-based information. Knowledge-based information consists of rules and other types of relations whereby designers can save their corporate know-how and reuse it later on to drive their design processes.

## Knowledge Inspector

An analysis tool which helps users understand how changing any property of their design (such as material, pressure, or a dimensional parameter) changes the operation or design of the product on which they are working. The Knowledge Inspector offers two options:

- "What if" to examine interactions of parameters with each other and with the rules that make up the product's specifications
- "How to" to see how a design can be changed to achieve a desired result

# M

## magnitude type parameter

A parameter whose value is defined by a quantity expressed in specific units. Length, Angle, Time parameters are magnitude type parameters. Boolean, Real, String and Integer parameters are not magnitude type parameters.

# P

## parameter

A feature defining a document property.

# R

## relation

A knowledgeware feature which, depending on certain conditions:

- sets parameter values
- displays a message
- or runs a macro.

Knowledgeware relations are formulas, checks, rules and design tables.

### **rule**

A set of instructions, generally based on conditional statements, whereby the relationship between parameters is controlled. In addition, depending on the context described by the rule instructions, CATIA macros can be executed and messages can be displayed. A rule is a feature. In the document specification tree, it is displayed as a relation that can be activated or deactivated. Like any feature, a rule can be manipulated from its contextual menu.

## W

### **wizard**

A form of user assistance that guides the user through a difficult or complex task within an application. The formula wizard helps the user typing formulas by picking up parameters either in the dictionary, or in the geometry area or in the specification tree.