Defining Constraint condition

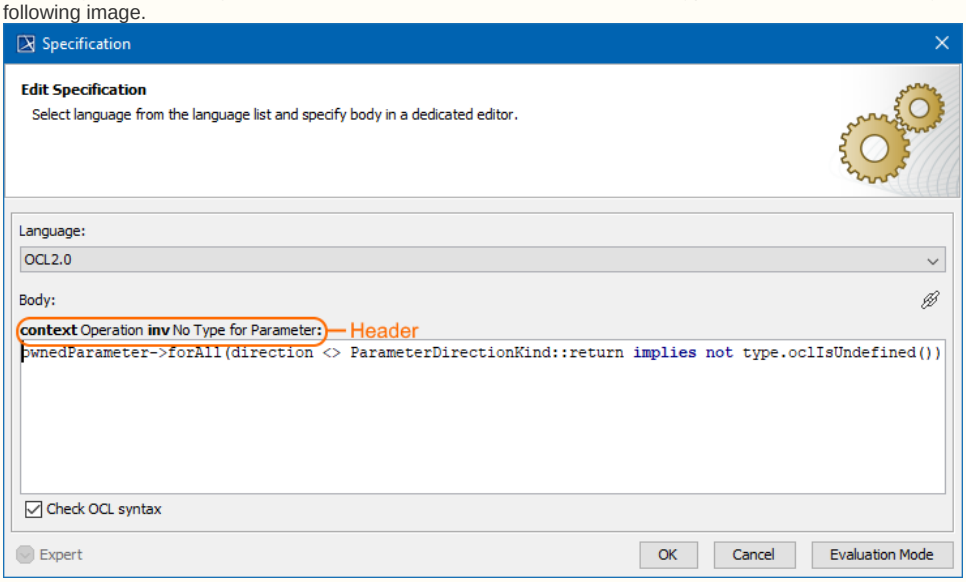
You must specify the validation condition for a validation rule. This condition must be true when evaluated in order for the Constraint to be satisfied. The result of a validation rule must be of the boolean type.

To define a Constraint condition

---

1. Open the Constraint Specification window. How to open the Specification window >>
2. Find the **Specification** property and select its value box.
3. Click  . The **Specification** dialog opens.
4. From the **Language** list, select one of the following languages:

- **OCL 2.0** is used for validation rules, specified in OCL language. Learn how to create OCL2.0 validation rule in Developer Guide >>

> **OCL Header**
>
> If you select the OCL2.0 language, the header of the expression from the constraint information is generated automatically according to the following rules: **context** <constrained element> <constraint type> <constraint name if any>:. See the following image.
>
> 
>
>
> **Constraint types**
>
> Since the Constraint is stereotyped by «validationRule» which is derived from «invariant » stereotype, **inv** is shown in the header. Only **inv** constraints can be evaluated. Other types of Constraints are not evaluated, but can be modeled for documentation purposes:
>
> - **def** – for the expression of the constraint with «definition» stereotype applied.
> - **init**, **derive** - for the expression of the default value of the property.
>
> > The **derive** expressions can be evaluated indirectly when the validation rule (**inv** constraint) is referencing the property and the validation rule is evaluated.
>
> - **pre**, **post**, **body** - for the expression of the appropriate fields of operation.
>
> **OCL Performance**
>
> When evaluating the validation rule defined in OCL language, the validation on the first run can have a delay of 20-30 seconds (depending on the computer performance) while the Java compiler is loading. Subsequent validations will run faster than the first one.
>
> If the validation process is run heavily on medium-large projects, increasing the default Java VM size is advisable. By default, the VM size is set to 400MB in MagicDraw; increasing this to 600 (or 800 if the computer has sufficient RAM) might improve the performance.

- **Binary** is used for more advanced expressions not easily expressed in OCL.These expressions are written in Java, compiled, and specified in the MagicDraw classpath. These expressions can then be specified as validation rule expressions. Learn more about binary validation rule in Developer Guide >>

- Scripts of **JavaScript**, **Jython**, **Groovy**, and **BeanShell**. Learn more about script writing in Creating executable opaque behaviors. Since constraints cannot have parameters, you may skip the information about managing parameters.
- **StructuredExpression**. Learn more about specifying criteria for querying model >>

5. In the **Body** box, type the expression of the selected language syntax.

> MagicDraw can evaluate only those validation rules whose expression is defined in one of the languages listed above. The other languages can be used only for documentation purposes.

> You can add hyperlinks in the **Body** box. How to define hyperlinks >>
>
> You can define expressions for global validation rules. Learn more about global validation rules >>

6. (optional) Click the **Evaluation Mode** button to to execute an expression on the actual testing model while editing. How to use the **Evaluation Mode** >>
7. Click **OK**.
   The Constraint condition is defined.