camos.

camos Develop
Developer training

Value attributes

camos.

Prerequisites

- **Knowledge base „Carconfigurator" at the end of the 3. day of the modeler training**

- **Contents:**
  - Displaying every color with preview graphic
  - Paintwork colors via num. Code with TextElement as naming
  - Access via function ValueAttributeGet()

camos.

Training targets

- **After these exercises you should …**
  - Name the advantage of the use of value attributes
  - Define and evaluate value attributes
  - Access value attributes via functions
  - Display value attributes on the form

camos.

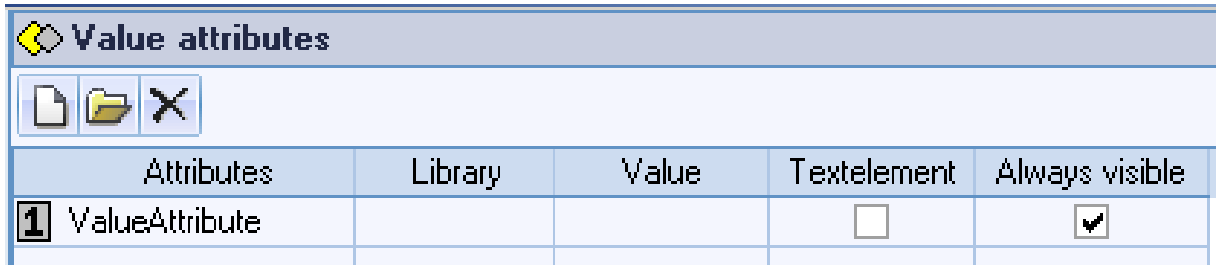
Value attributes

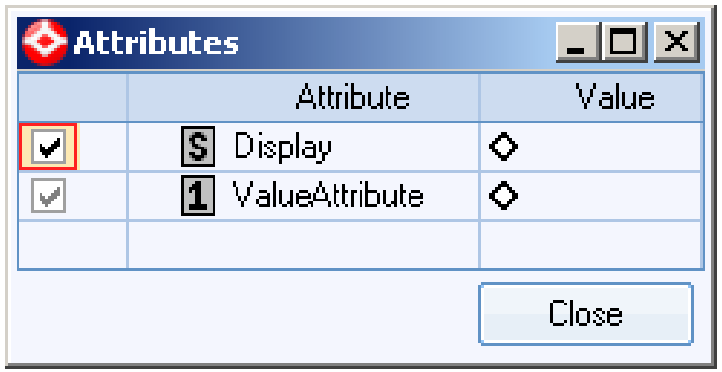
- **Value attributes**
  - are used to allocate further information to values of features
  - are defined in the knowledge base properties
  - Data types: Numerical, String, RTF, Date, Currency, Graphic, or HTML
    - The data type of the attribute does not have to concur with the data type of the feature!

- **Value of the attribute can be preset and overloaded on each individual feature value**

Value attributes

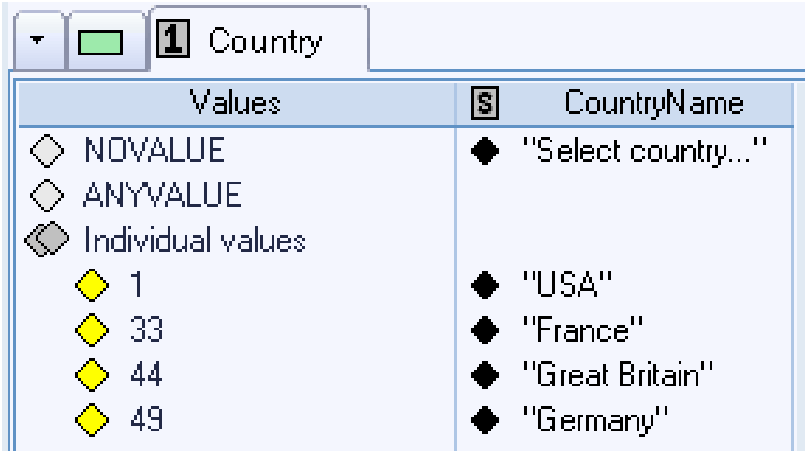
- **Display and definition**
  - Value attributes are displayed in the feature editor
    - if they are „always visible"
    - if they were defined in the editor as being visible (context menu „Define attributes" of the value table)





Value attributes

- **Usage on the form**
  - On many form elements can be set if the value itself and/or a value attribute is displayed

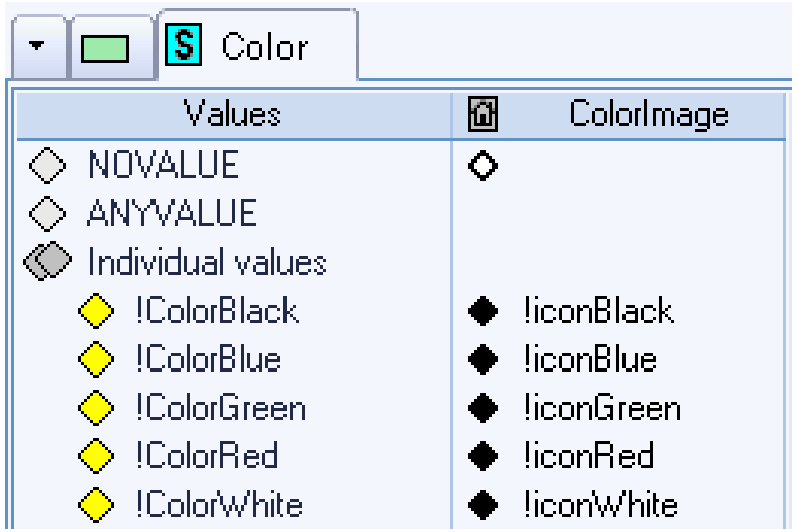


Exercise: Graphic preview

- **Paintwork color has to be displayed graphically**
  - Create graphic constants for each color (blue, red, green, black, white) in „Paintwork"
    - 16x16 pixel, fill in the corresponding color
  - Create value attribute
    - Name: „ColorImage"
    - Data type: Graphic
  - Show attribute „ColorImage" in the feature editor
    - Context menu „Define attributes" in value list of „Color"
  - Allocate graphic constants to the values

Exercise: Graphic preview

- **Show value attribute in form**
  - On the configbox Feature you create a new attribute column for the color and select the attribute „ColorImage"
    - Don't forget the overloaded forms in „Golf" and „Passat"!



- **Test the display during runtime**

camos.

Value attribute „Display“

- **Problem**
  - Working with multilingual values may become confusing and error-prone
  - Changes on the contents of multilingual texts can have undesired effects on the ruling and the loading of saved configurations

- **Solution: Using monolingual values**
  - The value is numerical or monolingual, e.g. code, contraction, identifier
  - The multilingual text is allocated via a value attribute

camos.

Exercise: Display-attribute

- **Target**
  - Depositing paintwork colors numerically, allocating the multilingual color descriptions as text elements

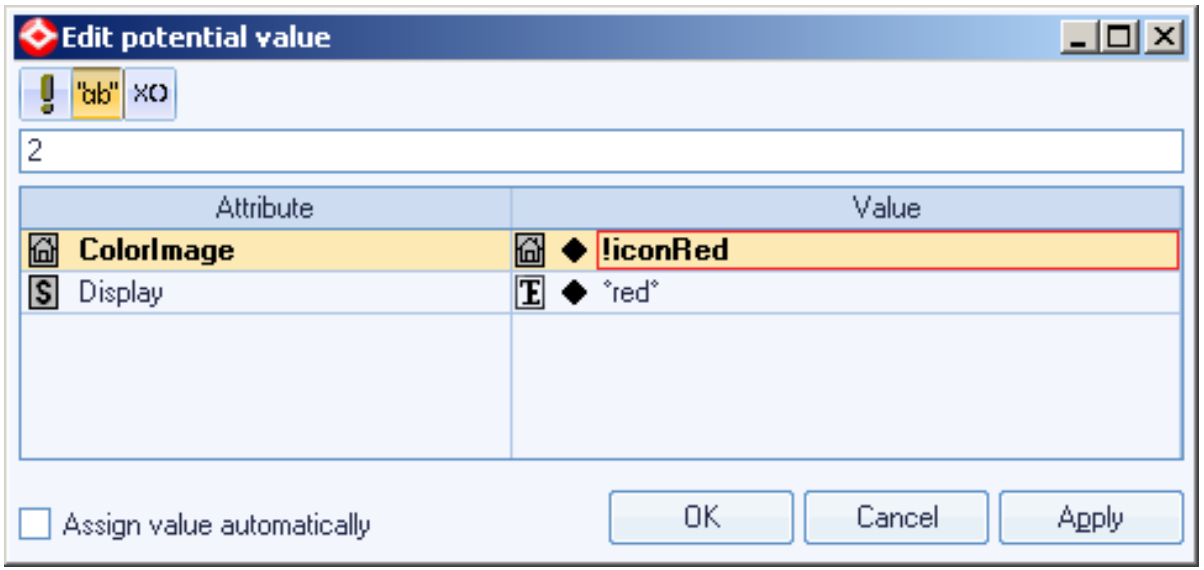
- **Create a further value attribute**
  - Naming: „Display"
  - Data type: String
  - Enable option „Text element"

- **Adapt the feature „Color" as follows**
  - Change data type from String to Numeric
  - Delete present potential values
  - Show attribute „Display" (context menu Value table)

camos.

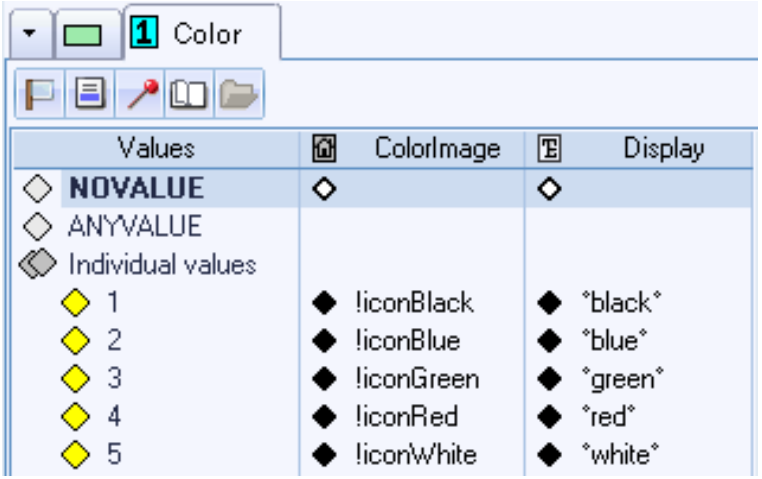
Exercise: Display-attribute

- **Further adaptations on feature „Color"**
  - Define numerical value for Color (1 to 5)
    - Maintain attributes „ColorImage" and „Display", assign value
  - Shift the rule under !PaintBlack to the new numerical value for „black"
  - Delete the string constants and the string values



Exercise: Display-attribute

- **The configbox should not show the numerical value but the value attribute „Display"**
  - Deposit the attribute „Display" in the field „Attribute of value"
    - Don't forget overloaded forms in „Golf" and „Passat"!



- **Even if the surface appears like in the previous exercise: internally the feature „Color" has the value 4, not „black"**

Exercise: Display-attribute

- **Problem: Result display shows color code**

Naming	Price
Otto engine 120	$800.00
205 tyres	$755.00
Alloy rims	$659.00
Metallic paintwork in 3	$980.00
Leather interior	$800.00
Radio	$605.00
Air conditioning	$1,020.00
CD-Changer	$600.00
Remote control key (1pcs.)	$60.00
Emergency key (1pcs.)	$20.00

- **Solution: SystemSet('RTF_ValueAttribute')**
  - Define that the value of the attribute „Display" has to be displayed with the result generation:

```
SystemSet('RTF_ValueAttribute', 'Display');
```

  - Features that do not have the attribute „Display" are of course still displayed with the feature value in the result

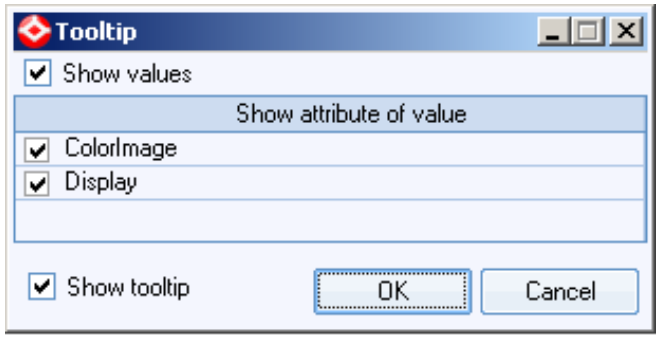
camos.

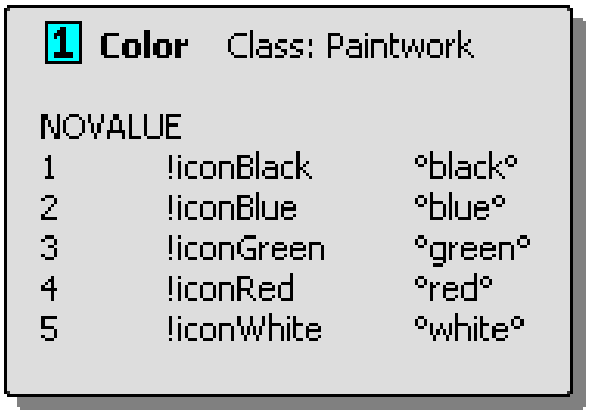
Access to value attributes

- **During runtime value attributes can be accessed via the following functions:**
  - ValueAttributeGet(Feature, Attribute[, Value])
    - determines the attribute value for a value
  - ValueAttributeValuesGet(Feature, Attribute, Value)
    - determines the values for an attribute value

Access to value attributes

- **Tip**
  - Tooltip help on a feature in the procedure editor shows the editor's values and attribute values
    - Click on the toolbar icon in the procedure editor
    - Define which information should be displayed





Exercise: ValueAttributeGet

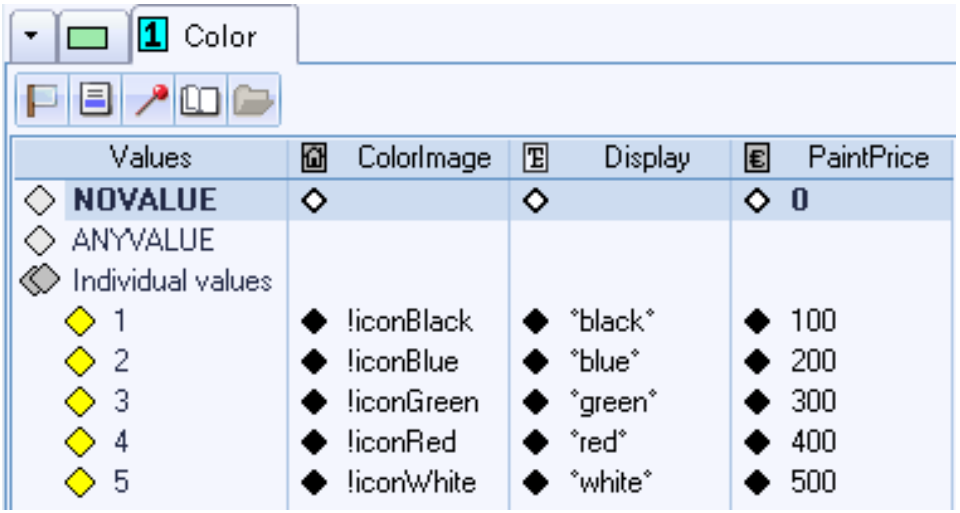
- **Target**
  - The individual colors have to be allocated with a price
  - This price has to be considered with the price calculation

- **Create a further value attribute**
  - Name: „PaintPrice"
  - Data type: Currency
  - Init value: 0

- **Fill „PaintPrice" with values**
  - Make the new attribute in the feature „Color" visible (context menu „Define attributes")
  - Deposit prices for the different colors

Exercise: ValueAttributeGet

- **Show prices**
  - Create a new attribute column on the configbox Feature of the „Color"
  - Don't forget the overloaded forms in „Golf" and „Passat"!



Exercise: ValueAttributeGet

- **Adapt price calculation**
  - Deposit assign trigger on feature „Color"

```
# deduct last price
@Car.ListPrice := @Car.ListPrice -
    ValueAttributeGet(Color, 'PaintPrice', LastValue);
# add new price
@Car.ListPrice := @Car.ListPrice +
    ValueAttributeGet(Color, 'PaintPrice');
```

Exercise: ValueAttributeGet

- **Adapt price calculation**
  - Overload Delete() in the class „Paintwork"
    - Local variable tmpPrice (currency)

```
IF ChangeTo = NOVALUE THEN
  tmpPrice := ValueAttributeGet(Color, 'PaintPrice');
  @Car.ListPrice := @Car.ListPrice - tmpPrice;
ENDIF;
@Car.ListPrice := @Car.ListPrice - Price;
```

- **Test the price calculation of the paintwork and color!**