genero®
report writer

BUILDING EXPRESSIONS

Using Genero Report Writer
GRS 3.00

Objectives

After this instruction, you will be able to:

- Understand the difference between RTL and PXML expressions
- Know what you can do with using RTL and PXML
- How to edit RTL and PXML
- Have an understanding on when you may need to use an RTL or PXML expression

Definition - RTL Expression

- A RTL expression is used to determine a property value **at run-time**
  - It is strongly typed and must return a type that the property is expecting
  - Examples
    - 'Text' property is a STRING, then the RTL expression should return a STRING
    - 'Visibility Condition' property is a BOOLEAN, then the RTL expression should return a BOOLEAN.
- Closely follows JAVA syntax
- Can use program variables as named in the DataView

RTL Operators

RTL uses the following operators:

- Arithmetic:

  + , **-** , ***** , **/** , **%** (modulus)

- String concatenation:

  +

- Relational/Boolean:

  <= , **<** , **>** , **>=**

  == (equal) , **!=** (not-equal) , **!** (not)

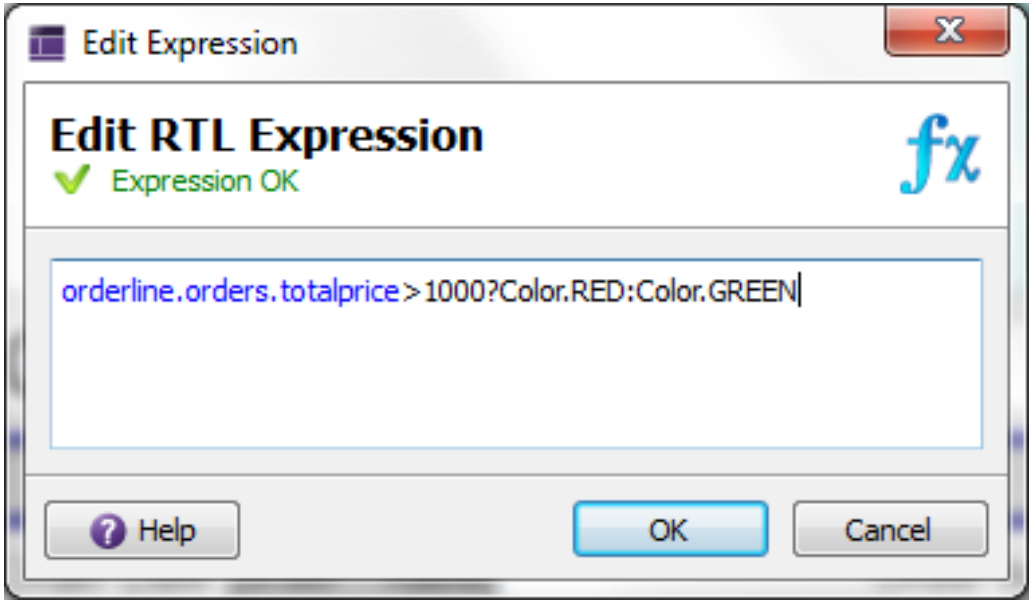
  && (and) , **||** (or)

RTL Conditional Operator

RTL uses the **?** & **:** conditional operators

- boolean-expression?value-if-true:value-if-false

```
value>=0?"Positive":"Negative"
```



Using program variables

- RTL can include program variables
  - Items available in the Data View
- Program variables have following member variables
  - **value**: Value of the program variable
  - **caption**: Title of the field
  - **name**: Name of the variable
  - **type**: RTL type of the variable
  - **isoValue**: locale and formatting independent value
- Typically found in value-related properties
  - Value property of DecimalFormatBox: fieldname.value OR fieldname

```
orderline.orders.totalprice.value
orderline.orders.totalprice
```

Definition

- RTL has following object classes
  - **Boolean**: all logical operations
  - **Color**: items related to color
  - **Enum**: list of values
  - **Date**: date formatting
  - **FGLNumericVariable**: program numeric Variables
  - **FGLStringVariable**: program non-numeric Variables
  - **Numeric**: all numeric operations
  - **Runtime**: change behavior at runtime
  - **String**: all string operations
- Each class has its own set of methods

Methods of String Class

- **charAt**(*Num index*)
- **contains**(*String*)
- **endsWith**(*String*)
- **equals**(*String*)
- **equalsIgnoreCase**(*String*)
- **format**(*String*)
- **indexOf**(*String*)
- **indexOf**(*String, Num index*)
- **isEmpty**()
- **isNull**()
- **lastIndexOf**(*String*)
- **lastIndexOf**(*String, Num index*)
- **length**()
- **matches**(*String*)
- **replace**(*String old, String new*)
- **replaceAll**(*String old, String new*)
- **replaceFirst**(*String old, String new*)
- **startsWith**(*String*)
- **startsWith**(*String, Num index*)
- **subString**(*Num startIndex, Num endIndex*)
- **substring**(*Num index*)
- **toLowerCase**()
- **toString**()
- **toUpperCase**()
- **translate**()
- **trim**()
- **trimCompress**()
- **trimLeft**()
- **trimRight**()
- **urlEncode**()

Examples:

```
orderline.orders.totalprice.toString()
orderline.orders.shipfirstname.trim().length()
```

String Methods

- Note with string methods
  - String index starts at 0 (as in Java)
  - String end index is the length
    (in other words, for the end index you count out starting with 1!)
  - substring(0,string.length()) returns the whole string

Methods of Numeric Class

- **abs**()
- **atan2**(*Numeric x*)
- **byteValue**()
- **cbrt**()
- **ceil**()
- **cos**()
- **cosh**()
- **exp**()
- **floor**()
- **format**(*"format-string"*)
- **getExponent**()
- **intValue**()
- **isInfinite**()
- **isNaN**()
- **isNull**()
- **log**()
- **log10**()
- **max**(*Numeric b*)
- **min**(*Numeric b*)
- **rint**()
- **round**()
- **signum**()
- **sin**()
- **sinh**()
- **sqrt**()
- **tan**()
- **tanh**()
- **toChar**()

Methods of Boolean Class

- Two values …
  - Boolean.TRUE
  - Boolean.FALSE
- A number will not be interpreted as a boolean value e.g. when defining the ‘Visibility Condition’ property
  - number.value (bad)
  - number.value==1 (good)

Color Class Methods and Static Members

- Static members
  - BLACK
  - BLUE
  - CYAN
  - DARK_GRAY
  - GRAY
  - GREEN
  - LIGHT_GRAY
  - MAGENTA
  - ORANGE
  - PINK
  - RED
  - WHITE
  - YELLOW

- Class Methods
  - **Color.fromHSBA**(*h, s, b*)
  - **Color.fromHSBA**(*h, s, b, a*)
  - **Color.fromRGBA**(*r, g, b*)
  - **fromRGBA**(*r, g, b, a*)
- Object Methods
  - **brighter()**
  - **darker()**
  - **getAlpha()**
  - **getBrightness()**
  - **getBlue()**
  - **getGreen()**
  - **getRGBA()**
  - **getHue()**
  - **getRed()**
  - **getSaturation()**
  - **toString()**

Enum classes and member variables

- Following properties have enumerated types
  - **Alignment**
  - **TextAlignment**
  - **BaselineType**
  - **LayoutDirection**
  - **X-SizeAdjustment**
  - **Y-SizeAdjustment**
  - **PageNoFormat**
  - **TrimText**
  - **Floating Behavior**
  - **Section**
  - **XYChartDrawAs**
  - **MapChartDrawAs**
  - **CategoryChartDrawAs**
  - **CodeType**
  - **BorderStyles**

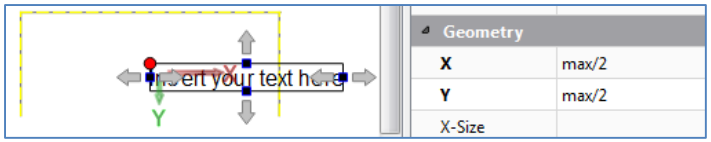
- Example of valid values for **Alignment** property
  - Baseline
  - Center
  - Far
  - Near
  - None
- Example of valid values for **TextAlignment** property
  - Center
  - Left
  - Right

Definition - PXML expression

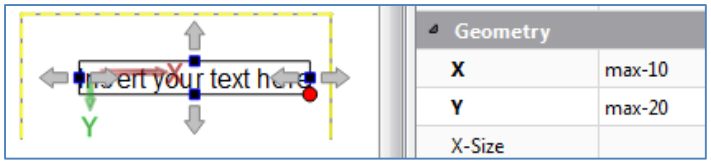
- A PXML expression is used to define the property of an object that is of PXML dimension type
  - co-ordinates and sizes
- Always yields a numeric value that can be measured in …
  - point|pt
  - pica|pc
  - inch|in
  - cm
  - mm
  - …
- Use units as a suffix (default units are points)
  - 10mm = 1cm
  - 1in = 72.27pt = 72.27

PXML built-in variables

- PXML has the following built-in **variables**
  - **min** – minimum extent of the current parent box
  - **max** – maximum extent of the current parent box
  - **rest** – remainder of the current parent box
- To center an object in its parent box
  - X=max/2
  - Y=max/2



- To position relative to right or bottom
  - X=max-10
  - Y=max-20



- To make an object take up the rest of its parent
  - Y=rest

PXML built-in functions

- PXML has the following built-in **functions**
  - **max**(*expression*, *expression*)
  - **min**(*expression*, *expression*)
  - **width**(*expression*) – the width required to print the expression in the current font
  - **length**(*expression*) – the length (height) required to print the expression in the current font
- Example: make an object wide enough to print the widest possible value from a 30 character field

  X-size=width("M")*30
- width(), length() return different values if font properties are different

Embedding RTL in PXML

- RTL can be embedded in PXML if it is delimited with {}
  - **width({"header.field.value"})**
  - **max({order_line.titlewidth}cm, width({"order_line.title"}))**
- Example: make an object wide enough to print the column header, and the widest possible value from a 30 character field

X-size=max(width("{"Order Id".translate()}"), width("M")*30)

RTL Syntax

- RTL syntax has following properties when compared to other languages
  - It is 1 line of code
    - fieldname.value.trim().length()
  - No intermediate/working variables
  - No loops

Common Uses

- String Concatenation

  header.firstname.trim()+" "+header.lastname.trim()

  – Print firstname followed by lastname, separated with a blank space

Common Uses

- Conditional values

  profit.value<0?Color.RED:Color.BLACK

  – Display negative profit values in a red font, otherwise use black font

Common Uses

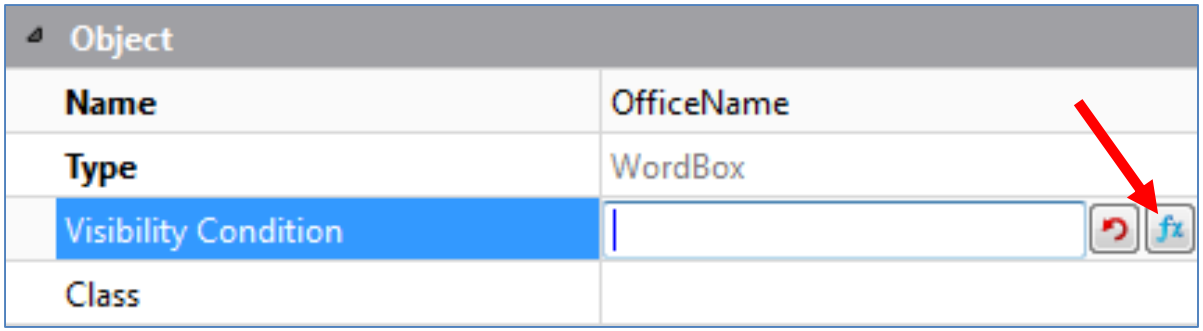
- Sizing fields

  max(width("{"label".translate()}"),width("M")*20)

  – Ensure that textbox width is wide enough to display both the caption applicable for the column, and 20 characters

Editing RTL and PXML

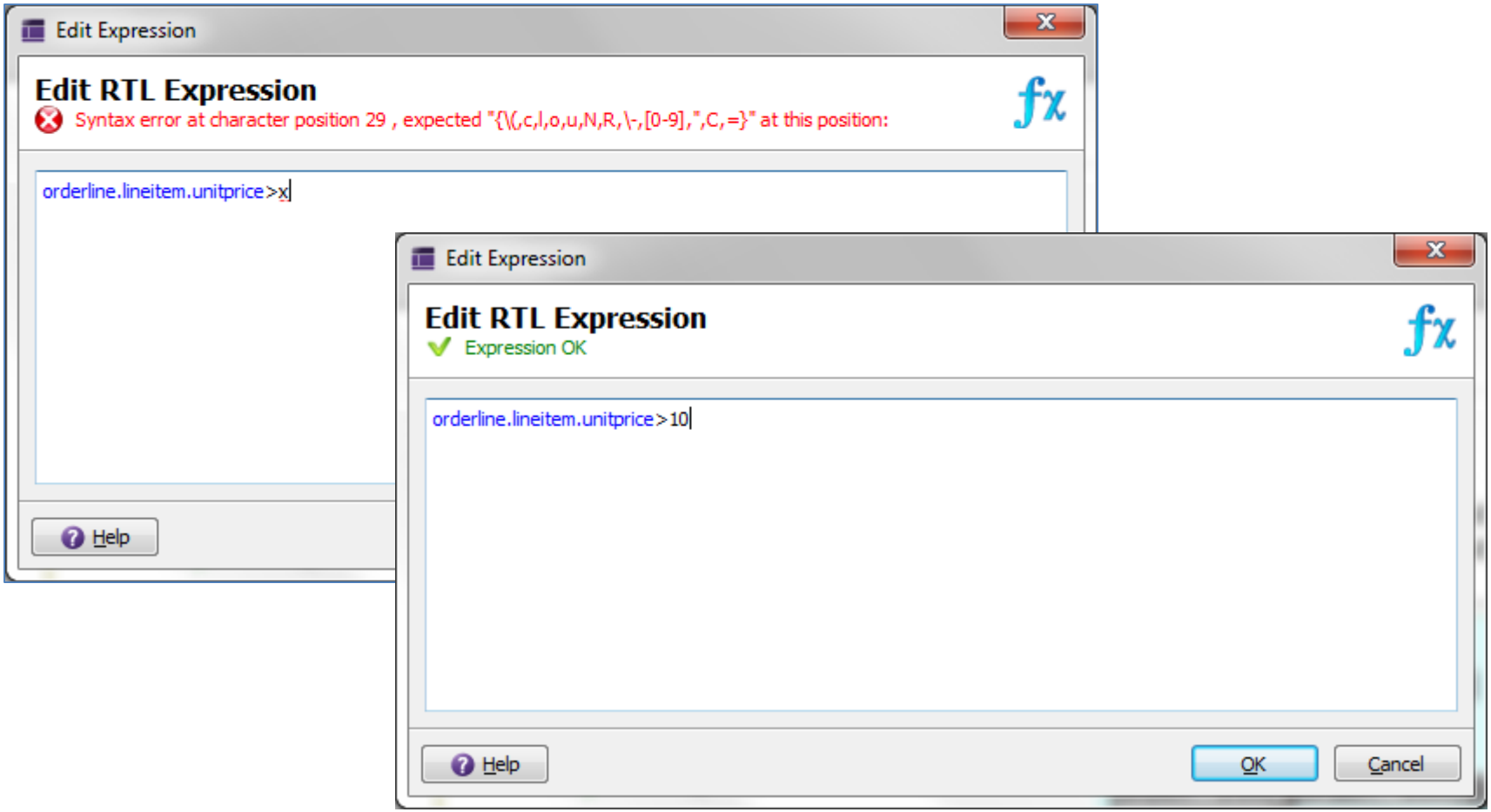
- Click on fx button
  - Place cursor in field to make fx button visible



- Warning! Other button restores to default value - take care!

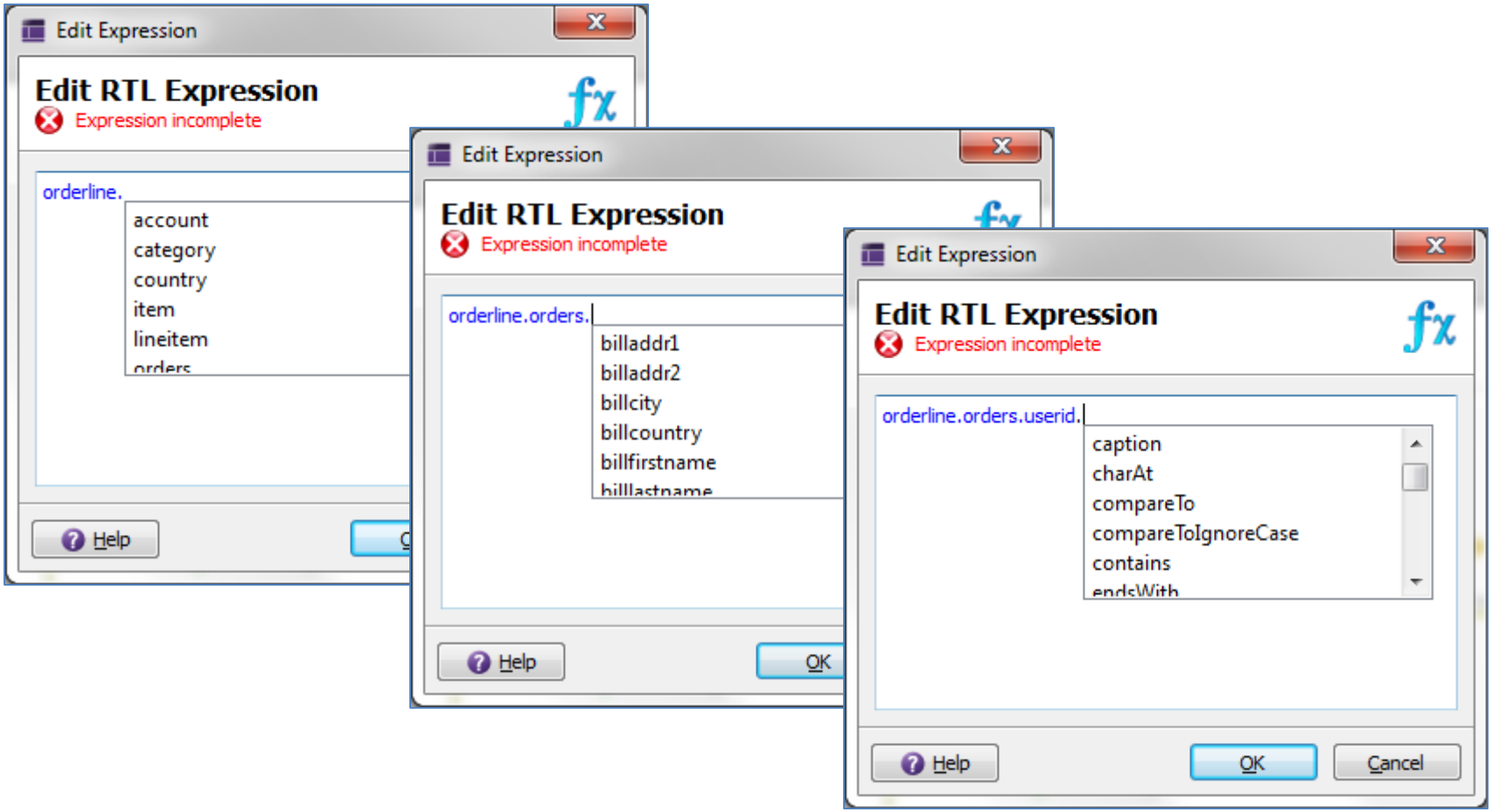
Editing RTL and PXML

Instant feedback



Editing RTL and PXML

- Control+SPACE = code-completion



When to use

- Where the property value is calculated at run-time, you need to use a RTL expression
- When determining a dimension type (co-ordinate or size), you need to use a PXML expression


QUESTIONS ?

Exercise

- Open the **'Sales' project**
- Change the title of the list report to display it in the format: "Sales at <shop_name> on <date>"
- Display the whole data line in bold if the price is lower than 20
- Change the display color of the Price
  - Display it in red if lower than 20
  - Display it in green if greater than 20
- Add a new column with the title "Rating" and with the value the image 'thumbs-up.jpg'
- Give the image a fixed size of 50 by 50 points
- Turn the image in the following way according to the value of 'Price':
  - If 'Price' is lower than 20, then turn the image upside down
  - If 'Price' is between 20 and 200, then turn the image to the right
  - If 'Price' is bigger then 200, keep the image unturned