Error Highlighting

The XSLT support can detect a range of errors in XSLT constructs, such as misspelled template names, missing template parameters, bad match-patterns, references to undeclared variables, wrong or useless embedded XPath expressions, etc. and also offers Quick Fixes to automatically fix some of those errors.

Syntax highlighting

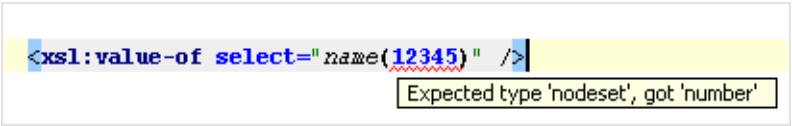
Where allowed, XPath function-calls, Axis names, Numbers, Strings, etc. are highlighted according to the currently active color scheme. By default, the plugin uses the colors that are defined for the corresponding Java types, such as number- and string literals, etc. If a different coloring is desired, those colors can be configured in **IDE-Settings | Colors & Fonts** on the tab **XPath**.

XPath syntax checks

Just like the interactive XPath Expression Evaluation, the XSLT support catches any syntax error in XPath expressions used inside a stylesheet.

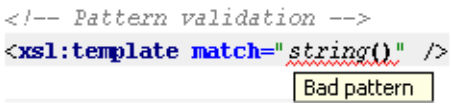
XPath type checking

In XPath, almost all types are assignable to each other with certain well-defined conversion semantics. However, the conversion to NODESET isn't defined for any type and there's also no (portable) conversion function available. Such type-conversions are highlighted as an error.



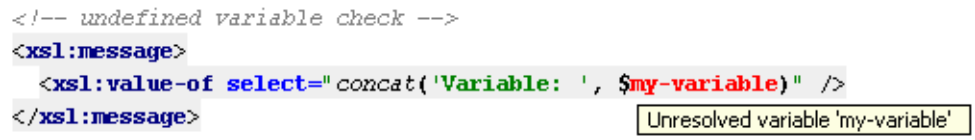
Pattern validation

A special form of XPath expression are patterns in XSLT. They are e.g. used as the value of the *match*-attribute in `xsl:template` elements. Only a certain subset of XPath expressions is allowed here, which the XSLT support checks for.

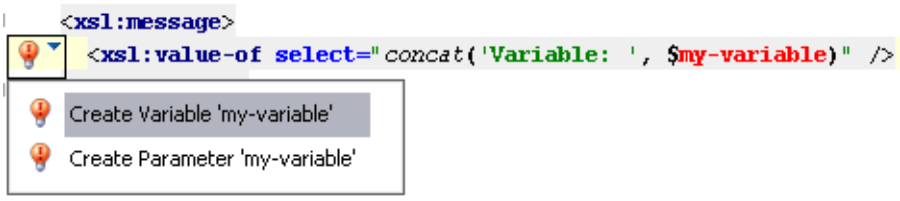


Unresolved references

References to variables that have not been declared or are not accessible from the current scope are detected and highlighted as an error. There are Quick Fixes available to create a variable or parameter declaration for such unresolved variables references.



Quick-Fixes:



Duplicate declarations

In XSLT there must not be more than one variable or parameter declared on the same scope level. It's also not allowed that there is more than one template with the same name. The plugin will identify such duplicate declarations and highlight them in the editor.

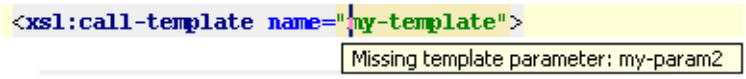
Other checks

Shadowed variables

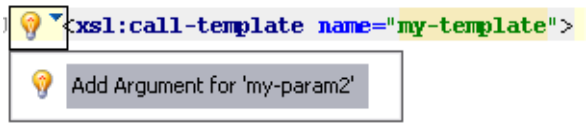
Even though is possible to have identically named variables or parameters in different nesting levels, this can be confusing and is likely to cause programming mistakes. The plugin can identify shadowed declarations and offers Quick-Fixes to either rename the local or the outer variable.

Missing template arguments

Another check that the XSLT support performs is whether all required parameters are specified with a `xsl:call-template`. A parameter is considered required if there is no default value, i.e. if there's no *select* attribute and the paremeter's declaring element has an empty body.

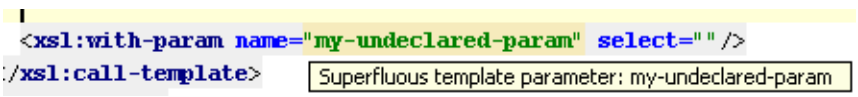


Quick-Fixes:

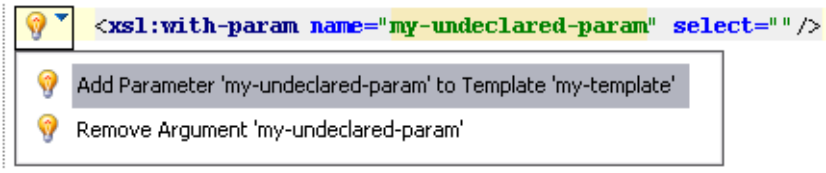


Superfluous template arguments

There's also a supplemental check that flags arguments that are not declared as template parameters. There are Quick Fixes available to either remove the argument from the template-call or to add a corresponding parameter to the called template.

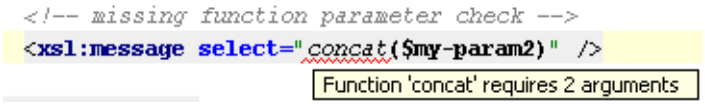


Quick-Fixes:



Function call arguments

The XSLT support does, just like the interactive XPath Expression Evaluation, check whether the number and types of function arguments match their declaration for built-in functions of XPath and XSLT.



XPath inspections

All XPath Inspections are supported for editing XSLT documents. Those inspections can also be suppressed in a way that is similar to the standard suppression-mechanism IntelliJ IDEA uses for Java code by using *noinspection* XML comments. The suppression is possible on different levels, either on instruction level, template-level (if applicable) or stylesheet level.