Base Language Extensions Style Guide

Base Language is by far the most widely extended language in MPS. Since it is very likely that a typical MPS project will use a lot of different extensions from different sources or language vendors, the community might benefit from having a unified style across all languages. In this document we describe the conventions that creators should apply to all Base Language extensions.

Quick Reference

If you use...	Set its style to...
.	Dot
[	LeftBracket
]	RightBracket
{	LeftBrace
}	RightBrace

= + - * / % ++ -- ! == != > >= < <= && || ?: (ternary) ~ << >> >>> & ^ | :eq: :ne:	Operator
	KeyWord

```
@interface
abstract
assert
boolean
break
byte
case
catch
char
class
const
continue
default
do
double
else
enum
extends
false
final
finally
float
for
get
goto
if
implements
import
instanceof
int
interface
long
native
new
null
package
private
protected
public
return
set
short
static
strictfp
super
switch
synchronized
this
throw
throws
transient
try
true
void
volatile
while
```

Keywords

A keyword is a widely used string, which identifies important concepts from a language. For example, all the primitive types from Base Language are keywords. Also names of statements such as ifStatement, forStatement are keywords. Use the KeyWo rd style from base language's stylesheet for keywords.

Curly braces

Curly braces are often used to demarcate a block of code inside of a containing construction. If you create an if-like construct, place opening curly brace on the same line as the construct header. I.e. use:

```
abc {
  code
}
```

instead of

```
abc
{
  code
}
```

Use the LeftBrace and RightBrace styles to set correct offsets. Make sure that the space between a character which is to left to opening curly brace and the curly brace itself is equal to 1 space. You can do so with a help of padding-left/padding-right styles .

Parentheses

When you use parentheses, set the LeftParen/RightParen styles to the left/right parenthesis. If a parenthesis cell's sibling is a named node's property, disable the first/last position of a parenthesis with first/last-position-allowed style.

Identifiers

When you use named nodes: methods, variables, fields, etc, it's advisable to make their name properties have 0 left and right padding. Making identifier declaration and reference holding the same color is also a good idea. For example, in Base Language , field declarations and references have the same color.

Punctuation

If you have a semicolon somewhere, set its style to Semicolon. If you have a dot, use the Dot style. If you have a binary operator, use the Operator style for it.

Previous Next