

Basics of the Action Script Language


The Action Script language comes with a [syntax scheme](#) as explained below. Also, the [reserved keywords](#) of the Action Script language cannot be used as variable names.

When writing action scripts, you should respect the following explanations regarding

- [Creating Objects of Base and Complex Types](#)
- [self Context](#)
- [Object References](#)
- [Guarded Statements](#)
- [Local Variables](#)
- [NULL Values](#)
- [Constructors](#)

Syntax Scheme

The following syntax scheme applies to the action script language.

Topic	Description	Examples																			
Names	All object and attribute names in Action Scripts must follow the syntax below: <div>[a-zA-Z] ([a-zA-Z0-9]) *</div> <p>However, UML names are not restricted and may contain any character. To use these names in Action Scripts apply backticks as follows:</p> <p>If the name of an element contains other characters, it must be escaped by backticks as follows:</p> <div>`invalid name`</div> <p>Consequently, the only characters not allowed in element names that are to be used in action scripts are backticks (`).</p> <div><div> If the invalid attribute name is given by an external representation, e.g. it has to be serialized into an XML document, flat file, IDocs or such, you can also use the attribute externalName on attributes to hold the external name. This way, you can use a valid name internally. Once the data is serialized, the external name will be used.</div></div>	Some escaping examples: <table><tr><th>Element</th><th>Name</th><th>Usage With Escaping</th></tr><tr><td rowspan="2">Class attribute</td><td>strange name having blanks and dots: ...</td><td>object. `strange name having blanks and dots: ...` ;</td></tr><tr><td>a-b</td><td>object. `a-b` ;</td></tr><tr><td>Operation</td><td>::my strange operation name::</td><td>object. `:my strange operation name::` `();</td></tr><tr><td>Parameter</td><td>äparam</td><td>set `äparam` = "test";</td></tr><tr><td rowspan="2">Object</td><td>a/b</td><td>create `a //b` ;</td></tr><tr><td>17_name</td><td>create `17_name` ;</td></tr></table>	Element	Name	Usage With Escaping	Class attribute	strange name having blanks and dots: ...	object. `strange name having blanks and dots: ...` ;	a-b	object. `a-b` ;	Operation	::my strange operation name::	object. `:my strange operation name::` `();	Parameter	äparam	set `äparam` = "test";	Object	a/b	create `a //b` ;	17_name	create `17_name` ;
	Element	Name	Usage With Escaping																		
Class attribute	strange name having blanks and dots: ...	object. `strange name having blanks and dots: ...` ;																			
	a-b	object. `a-b` ;																			
Operation	::my strange operation name::	object. `:my strange operation name::` `();																			
Parameter	äparam	set `äparam` = "test";																			
Object	a/b	create `a //b` ;																			
	17_name	create `17_name` ;																			
Case Sensitivity	All operations, statements, variable names, class names, and attribute names are case sensitive.	<ul style="list-style-type: none">• The variables myVar and myvar are two different variables.• The xUML Compiler will not recognize an assignment statement <code>Set.</code> you need to write <code>set</code> in lowercase instead.																			

On this Page:

- [Syntax Scheme](#)
- [Object Navigation](#)
- [Reserved Keywords](#)

Related Pages:

- [Creating Objects of Base and Complex Types](#)
- [self Context](#)
- [Object References](#)
- [Guarded Statements](#)
- [Local Variables](#)
- [NULL Values](#)
- [Constructors](#)

Scripting Style	Operations can be scripted in an object-oriented or a procedural syntax style. The differences are shown in the example table on the right.	Object-Oriented Style	Procedural Style
		myVariable. exists();	exists (myVariable);
		myDataItem. myAttr. exists();	exists (myDataItem. myAttr);
		s.substring (0, 5);	substring(s, 0, 5);
		set aString = s.substring (0, 5). toUpper();	set aString = toUpper (substring (s, 0, 5));

Object Navigation

Use the following syntax to navigate to properties and operations within an object:

Target	Syntax	Example
Property	objectName.propertyName	product.category
Operation	objectName.operationName (parameters)	product.new()
Sub-property of a complex property	objectName.propertyName. subPropertyName	product.storage. area
Array property	objectName.arrayProperty	product.versions
Array property element	objectName.arrayProperty[index]	product.versions [3]

Reserved Keywords

The following keywords, constants, operators etc. should not be used as variable names as they are reserved for action script execution:

- a
- nd
- a
- p
- p
- e
- nd
- a
- p
- p
- ly
- a
- rr
- ay
- by
- c
- a
- t
- ch
- c
- r
- e
- a
- te

- d
i
s
ti
n
ct
- e
a
ch
- e
l
se
- e
rr
o
r
Code
- e
rr
o
r
T
y
pe
- f
a
l
se
- fi
r
st
- fr
om
- g
r
o
up
- if
- l
a
st
- li
ke
- l
o
c
al
- NULL
- or
- r
e
d
u
ce
- s
e
l
e
ct
- set
- single
- sort
- then
- to
- true
- unlike
- use
- using
- where