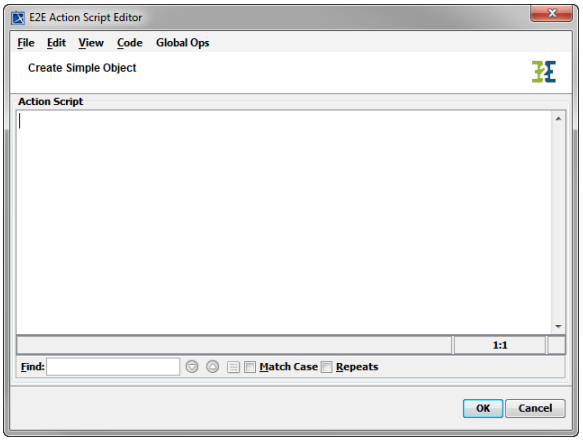
Using the Suggestion Features of the Action Script Editor

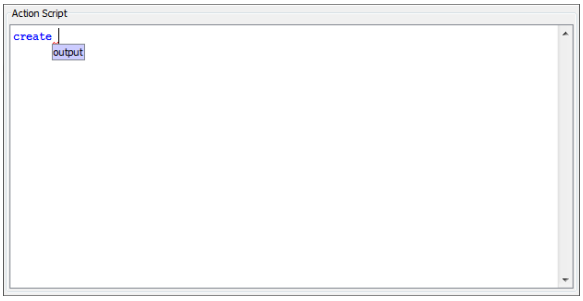
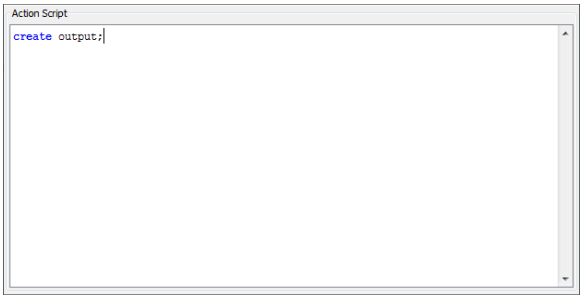
The **Action Script Editor** dialog opens.

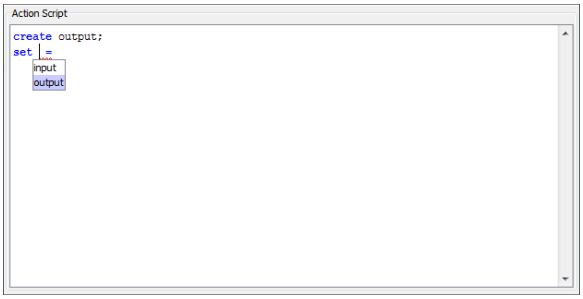
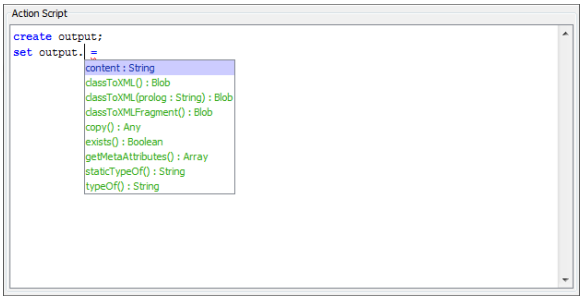
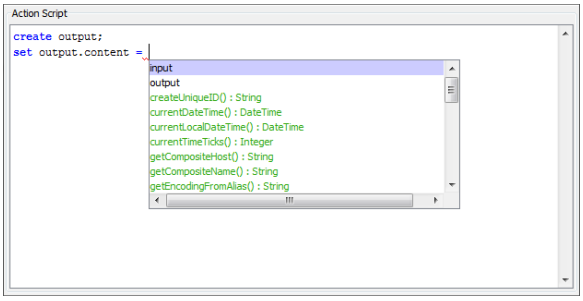
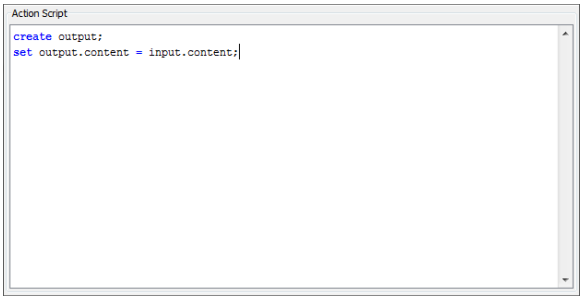


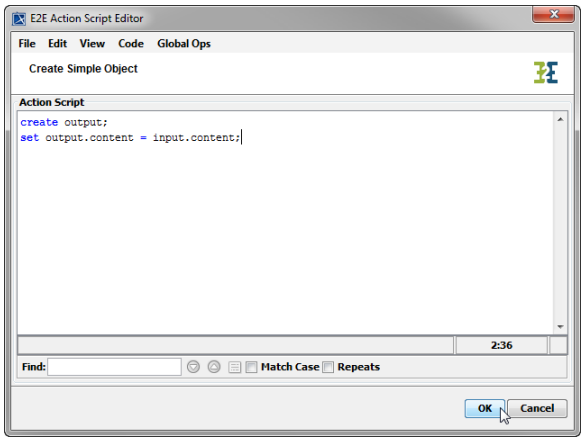
On this Page:

The xUML Action Language is **case sensitive**. Every statement must end with a **semicolon**. Operations, functions, and macros of the action language are displayed in **blue**.

Entering a First Action Script Statement

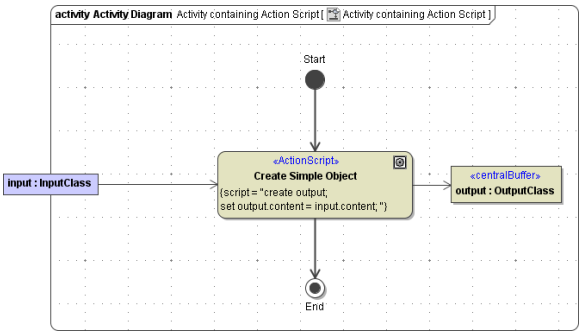
	To create an action script statement, for instance a `create` assignment statement, start by typing the create keyword. After adding a space, all available object nodes of the action node are listed.
	Add a semicolon to finish the statement.

	Now try the set statement to assign a value to the property of the newly created object. The Action Script Editor automatically adds an equal sign (=). Select the object output from the suggestion list by clicking it. Alternatively, use the up and down arrow keys and confirm a selection by pressing **Enter**.
	After adding a dot, all properties and operations of the object are listed in the suggestion list. Select the property content by clicking or use the keyboard. You can use the mouse or the keyboard to select an object, an operation, etc. from the list of suggestions provided by the Action Script Editor. You can also open this list manually by pressing **Ctrl** together with **Space**.
	Move the cursor to the end of the line and force the suggestion list by pressing **Ctrl** + **Space**.
	Finalize the statement by selecting the property content of object input and adding a semicolon.



The action script for creating an object and assigning a value to a property is complete.

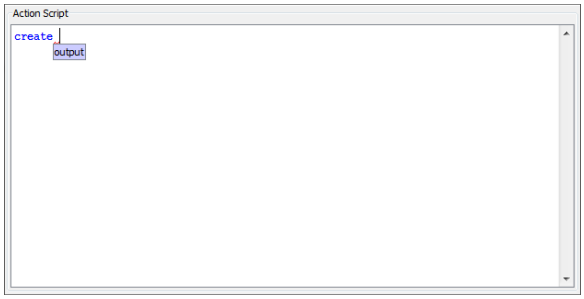
Click **OK** to save the changes and close the editor.



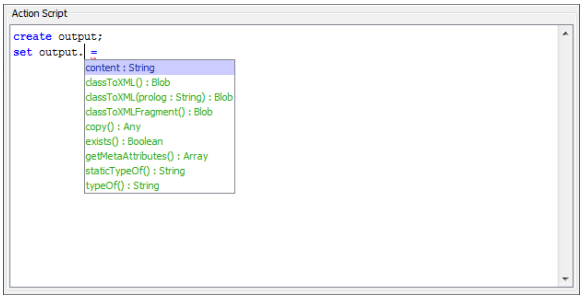
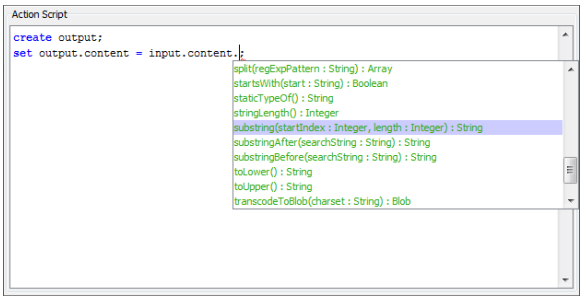
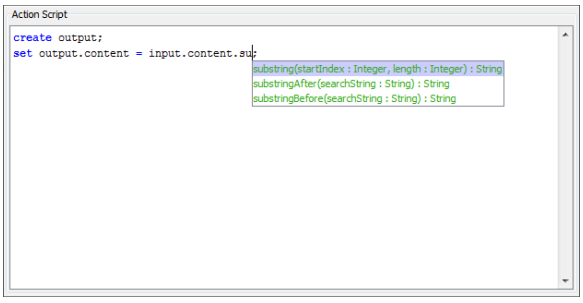
You can recognize an action using an action script by the <<ActionScript>> stereotype and the tagged value **script** (see entry `{script="..."}` in the picture above).

Overview on the Suggestion List Features

As shown in the example above, the Action Script Editor helps you by offering appropriate elements that can be used in the action script statement.



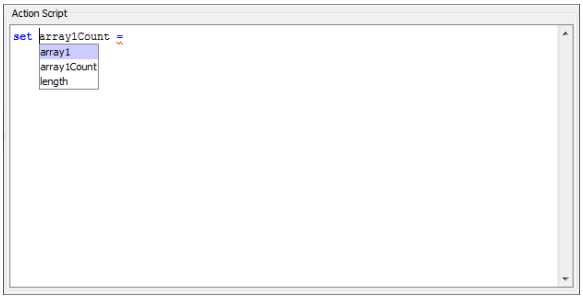
Writing a create statement (e.g. `create output;`, the Action Script Editor offers all output object nodes defined for the action node in the activity diagram.

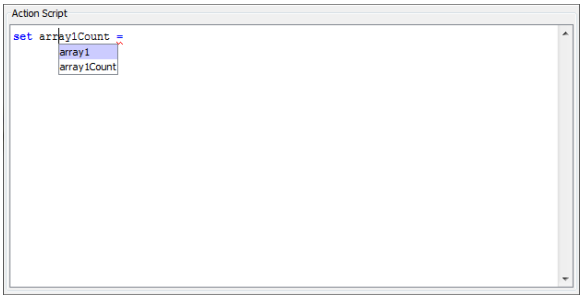
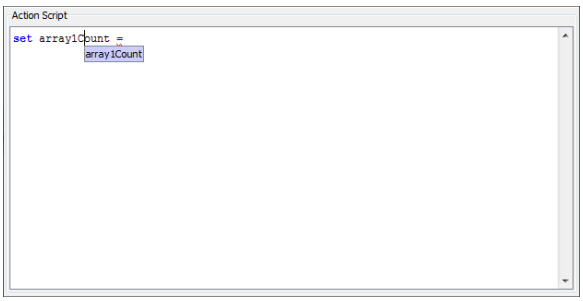
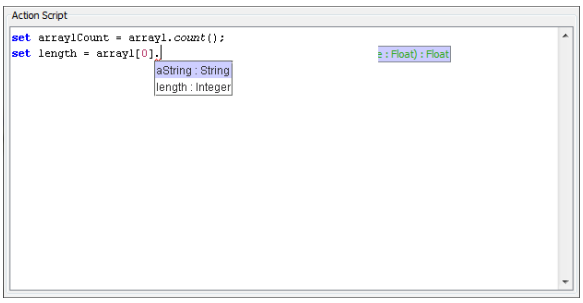
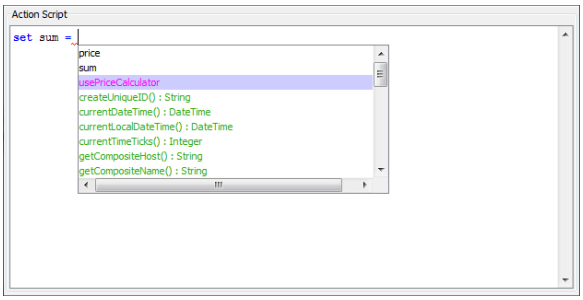
	Referencing an object or an object property (e.g. `output` or `output.content`), the Action Script Editor offers all properties and operations defined for the class when the cursor is immediately after the dot following the object name. The Action Script Editor colors object properties in black, attributes of a complex type in blue, and operations in green.
	Referencing the operations of an object, the Action Script Editor offers each operation with its corresponding parameters.
	The suggestion list can be filtered by starting to type the initial letters of the operation. The filtering is case insensitive.

Forcing the Suggestion List

Forcing the suggestion list by pressing **Ctrl** + **Space**, the Action Script Editor offers a filtered list dependent on the position of the cursor within the object node name. The Action Script Editor offers only objects, properties and operations that match the filter.

For example, in an activity diagram with object nodes: `array1`, `array1Count`, and `length`:

	If the cursor is placed before the string "array1Count", the filter is empty and the Action Script Editor offers all 3 object nodes.

	If the cursor is after "arr", the filter contains "arr" and the Action Script editor offers only a `rray1` and `array1Count`.
	If the cursor is after "array1C", the filter contains "array1C" and the Action Script editor offers only a `rray1Count`.
	Referencing an array object node (e.g. `array1`), the Action Script Editor only offers operations that are relevant for arrays.
	Referencing an array element object (e.g. `array1[0]`), the Action Script Editor offers all attributes defined for the array element class.
	Static operations of classes can be made available by use-dependencies. These use-dependencies are listed in pink in the suggestion list.

Action Script
`set sum = usePriceCalculator:`
`calculatePrice(price : Float, exchangeRate : Float) : Float`
	After selecting the use-dependency, add a colon to get a suggestion list of all operations that are available via this dependency.