Format file description

A format file is defined for each scheduling entity type: SMART Table, Sub Table, and Job.

When writing a draft each entity will be written according to the instructions provided in the corresponding format file.

Element and attributes will be written in the same order they appear in the format file.

Each line in the format file is of the following form:

<Identifier> <type>: internal\_name=<value> draft\_name=<value> [<key>=<value> <key>=<value>…]

Use ‘#’ to remark lines, if not used in beginning of a line must be preceded by a whitespace.

**Identifier** is a logical value used to group lines together into elements, its value is not reflected in the output.

**Type** declares to what the format is applied. The value *element:* declares a line describing an element tag, the value attribute declares a line describing an attribute.

**internal\_name** The element or attribute name as it is referred to within the conversion tool (usually in Globs.java).

**draft\_name** The element or attribute name as it should be written to the draft.

**Key=value** pairs provide additional formatting, the key must be one of the keys described below. There must not be any whitespaces between the key and the ‘=’ sign. There are only to constraints for value:

1. Cannot contain newline.
2. ‘=’ characters must be escaped by a backslash (‘\’).

The following keys may be used to define additional formatting:

**default** used to set default value for mandatory attributes if SchedEntity implementer returned null for them.

**mandatory** flags an attribute as mandatory, to be written even if null value was returned for it.

**transformation** define transformations to be performed on value.  
Transformations are defined as a list of function calls – function(arg1, arg2, … argn)  
Both function and args are case sensitive.

Double quotes will be stripped of args ("foobar" 🡪 foobar), commas and parenthesis in values should be escaped by backslash.  
If more than one transformation is used, separate with spaces: *func1() func2(arg)*   
All functions will be called from left to right. Transformed value will be passed to the next call  
Transformation functions:

* **msg\_length()** prefix the value with its length in 5 digits (used for domail messages).
* i**nterval\_unit()** change from a number to the same in 5 digits postfixed with “M” if number was provided with unit (e.g. “7H”), will preserve unit and change number to five digits.
* **replace (str1, str2)** replaces all occurrences of str1 with str2.   
  Will call Java’s String.replace(str1,str2)
* **replace\_regex (regex, str)** replaces all regex’s matches with str.   
  Will call Java’s String.replaceAll(regex,str)
* **substring (begin, end)** changes values to the substring starting at index begin, and ending at index end-1  
  Will call Java’s String.substring(begin,end).

**subelements** declare an external format file from which this element’s subelements are to be loaded.  
Currently used only for ON statements.

**condition** a logical expression, attribute will be written only when condition is met (takes precedence over **mandatory**).

Each part of the condition states an attribute, and operator and a value.

Attribute is an attribute of the sched entity (e.g. CYCLIC).  
 operator is one the following:

* == met if attribute’s value is the same as expressions value.
* != met if attribute’s value is not the same as expressions value.
* =~ met if attribute’s value matches an expression
* !~ met if attribute’s value does not match an expression

Relationship between expressions is defined using ‘OR’ or ‘AND’

Example:

‘condition=jobname\=~.+z AND CYCLIC\=\=1’ is met by cyclic jobs whose name ends in ‘z’

When evaluation condition ‘OR’ takes precedence over ‘AND’.

“A AND B OR C” will be met if A is true, as well as B or C. (A AND (B OR C))

Once all formats have been loaded, format overrides will be loaded according to version.

All overrides lower or equal to the version will be loaded in ascending order (7 🡪 8 🡪 9)

Overrides format:

<ENTTIY> element: ENTITY[/<element>[/<sub element>…] [attribute:<name>] <key=value>…

**<ENTITY>** declares to which entity the override applies must be SMART SUB or JOB. If applies to more than one entity, use commas to separate entities (e.g. SMART,JOB).

**element:** provide a full path to the overridden element. Must start with “ENTITY”, use slashes to append subelements. Use only internal names (as defined above) when providing element.

**attribute:** the overridden attribute’s internal name. If not provided overrides will be applied to element

**key=value** pairs (using the keys described above) provide the overrides.

If the attribute or element declared is not found, it will be added.

Examples:

To change ODATE attribute to ETADO for SMART Table and job in-conditions use the line:

SMART,JOB element: ENTITY/INCOND attribute: ODATE draft\_name=ETADO

To add a new mandatory attribute “foo” with draft name “foo”, and default value “bar” to sub tables use the line:

SUB element: ENTITY attribute: foo draft\_name=foo default=bar mandatory=true