X-DEVICE syntax

This page contains the syntax of the X-DEVICE deductive rule language in BNF notation. Notice that the grayed expressions indicate the extensions that X-DEVICE introduced to the language of DEVICE.

```
<rule> ::= if <condition> then <consequence>
<condition> ::= <inter-object-pattern>
<consequence> ::= {<action> | <conclusion> | <derived_attribute_template>}
<inter-object-pattern> ::= <condition-element> ['and' <inter-object-pattern>]
<inter-object-pattern> ::= <inter-object-pattern> 'and' <prolog_cond>
<condition-element> ::= ['not'] <intra-object-pattern>
<intra-object-pattern> ::= [<inst_expr>'@']<class_expr>['('<attr-patterns>')']
<attr-patterns> ::= <attr-pattern>[','<attr-patterns>]
<attr-pattern> ::= <attr-expr>['.'<path_expr>] {':'<variable> | <predicates>
                          | ':'<variable> <predicates>
                          | <list-operator> <variable>}
<path_expr> ::= <nt-attr-expr> ['.'<path_expr>]
<attr-expr> ::= {<nt-attr-expr>|<t-attr>|<normal-attr>'↑'}
<nt-attr-expr> ::= <nt-attr>[{'*'|<integer>}]
<nt-attr-expr> ::= {'*'|'+'}
<nt-attr> ::= {<normal-attr>|<system-attr>}
<t-attr> ::= {<xml-attr>|<empty-attr>}
<normal-attr> ::= <attr>
<system-attr> ::= '!'<attr>
<xml-attr> ::= '^'<attr>
<empty-attr> ::= '∅'<attr>
<attr> ::= {<attribute>|<variable>}
<predicates> ::= <rel-operator> <value> [{ '&' | ';' } <predicates>]
<predicates> ::= <set-operator> <set>
<rel-operator> ::= '=' | '>' | '>=' | '=<' | '<' | '\=' | '$' | <date-operator>
<date-operator> ::= '$'{'y'|'m'|'d'}
<set-operator> ::= '⊂' | '⊄' | '⊆' | '∈' | '∉' | '⊃' | '\⊃' | '⊇'
<list-operator> ::= 'ↄ' | '\ↄ'
<list-operator> ::= 'ↄ'<order_expr>
<order_expr> ::= {<abs_order>|<rel_order>|'{'<rel_order>','<abs_order>'}'}
<abs_order> ::= <rel-operator><integer> | <integer>
<rel_order> ::= { 'before' | 'after' }'('<variable>')'
<rel_order> ::= 'between' '('<variable>','<variable>')'
<value> ::= <constant> | <variable> | <arith_expr>
<set> ::= '['<constants>']'
<prolog_cond> ::= 'prolog' '{'<prolog_goal>'}'
<action> ::= <prolog_goal>
<conclusion> ::= <derived_class_template>
<conclusion> ::= {'xml_result' | 'shallow_result'}'('<elem_expr>')'
<conclusion> ::= {'xml_sorted' | 'shallow_sorted'}'('[<group_list>
                                                    ['-'<order_list>]','<elem_expr>')'
<elem_expr> ::= <derived_class_template>
<elem_expr> ::= <derived_class>'('<derived_class_template>')'
<derived_class_template> ::= <derived_class>'('<templ-patterns>')'
<derived_attribute_template> ::= <variable>'@'{<class>}'('<templ-patterns>')'
<templ-patterns> ::= <templ-pattern> [',' <templ-pattern>]
<templ-pattern> ::= {<normal-attr>|<system-attr>|<xml-attr>}':'{<value> | <aggregate_expr>}
<templ-pattern> ::= <empty-attr>
<aggregate_expr> ::= <aggregate_function>'('<variable>')'}
<aggregate_expr> ::= 'ord_list('<variable>['-'<group_list>['-'<order_list>]]')'
<aggregate_function> ::= 'count'|'sum'|'avg'|'max'|'min'|'list'|'string'
<group_list> ::= '['<variable>[','<variable>]']'
<order_list> ::= '['<ord_symbol>[','<ord_symbol>]']'
<ord_symbol> ::= {'<' | '>'}
<inst_expr> ::= {<variable>|<class>}
<class_expr> ::= {<variable>|<class>}
<class_expr> ::= <inst_expr>'/'<class>
<class> ::= an existing class or meta-class of the OODB schema
<derived_class> ::= an existing derived class or a non-existing base class of the OODB schema
<attribute> ::= an existing attribute of the corresponding OODB class
<prolog_goal> ::= an arbitrary Prolog/ADAM goal
<constants> ::= <constant>[','<constants>]
<constant> ::= a valid constant of an OODB simple attribute type
<variable> ::= a valid Prolog variable
<arith_expr> ::= a valid Prolog arithmetic expression
```