

Funcons-beta: Booleans *

The P_LanCompS Project

Booleans.cbs | PLAIN | PRETTY

Booleans

```
[ Datatype  booleans
   Alias    bools
   Funcon   true
   Funcon   false
   Funcon   not
   Funcon   implies
   Funcon   and
   Funcon   or
   Funcon   exclusive-or
   Alias    xor ]
```

Datatype booleans ::= true | false

Alias bools = booleans

Funcon not(_ : booleans) : ⇒ booleans

not(B) is logical negation.

Rule not(false) ⇔ true

Rule not(true) ⇔ false

Funcon implies(_ : booleans, _ : booleans) : ⇒ booleans

implies(B_1, B_2) is logical implication.

Rule implies(false, false) ⇔ true

Rule implies(false, true) ⇔ true

Rule implies(true, true) ⇔ true

Rule implies(true, false) ⇔ false

Funcon and(_ : booleans*) : ⇒ booleans

*Suggestions for improvement: plancomps@gmail.com.
Reports of issues: <https://github.com/plancomps/CBS-beta/issues>.

`and(B, \dots)` is logical conjunction of any number of Boolean values.

Rule `and()` \rightsquigarrow `true`

Rule `and(false, $_*$: boololeans*)` \rightsquigarrow `false`

Rule `and(true, B^* : boololeans*)` \rightsquigarrow `and(B^*)`

Funcon `or($_$: boololeans*)` : \Rightarrow `boololeans`

`or(B, \dots)` is logical disjunction of any number of Boolean values.

Rule `or()` \rightsquigarrow `false`

Rule `or(true, $_*$: boololeans*)` \rightsquigarrow `true`

Rule `or(false, B^* : boololeans*)` \rightsquigarrow `or(B^*)`

Funcon `exclusive-or($_$: boololeans, $_$: boololeans)` : \Rightarrow `boololeans`

Alias `xor` = `exclusive-or`

`exclusive-or(B_1, B_2)` is exclusive disjunction.

Rule `exclusive-or(false, false)` \rightsquigarrow `false`

Rule `exclusive-or(false, true)` \rightsquigarrow `true`

Rule `exclusive-or(true, false)` \rightsquigarrow `true`

Rule `exclusive-or(true, true)` \rightsquigarrow `false`