MODULE *fenestrate*

EXTENDS *Naturals*, *FiniteSets*, *TLC*

CONSTANTS
DAYS, here specified by integers
TIMES, here specified by integers as well
MIN_WINDOWS,
MIN_EXCLUSIONS

Selectors are functions which map each day to a boolean value representing whether or not a day matches a window

$SELECTORS \triangleq [DAYS \to \text{BOOLEAN}]$

This is the set of all possible windows in tuple form, where each tuple represents
- A selector, which is whether or not this window is active on a given day
- A start time
- An end time. If the end time is less than the start time, this window spans the next day.

$WINDOWS \triangleq SELECTORS \times TIMES \times TIMES$

All possible datetimes

$DATETIMES \triangleq DAYS \times TIMES$

Whether or not the given date/time falls within the window specified by the given window. This requires that the window's selector match the given day and that the *datetime*'s time falls after the start time ... and
- If the end time is greater than the start time, then ... and before the end time
- If the end time is less than the start time, representing a window extending to the next day, then ... OR before the start time

Note that a window is defined by the day on which its start time exists, so for example a window (24-hour day) starting at 23 and ending at 2 would NOT match a time of 1, although the previous day's window might.

$in\_window(window, datetime) \triangleq$
LET
$selector \triangleq window[1]$
$day \triangleq datetime[1]$
$time \triangleq datetime[2]$
$from \triangleq window[2]$
$to \triangleq window[3]$
IN
$\land\ selector[day]$
$\land\ from \leq to \Rightarrow\ \land\ from \leq time$
$\land\ time \leq to$
$\land\ from > to \Rightarrow\ \lor\ from \leq time$
$\lor\ time \leq to$

$in\_nonexcluded\_windows(windows, exclusions, now) \triangleq$

$\land \exists\, w \in windows : in\_window(w,\, now)$
$\land \neg\exists\, e \in exclusions : in\_window(e,\, now)$

--algorithm *fenestrate*

test
variables
now $\in$ *DATETIMES*,
windows = CHOOSE $w \in$ SUBSET *WINDOWS* : *Cardinality*(w) $>$ *MIN_WINDOWS*,
exclusions = CHOOSE $e \in$ SUBSET *WINDOWS* : *Cardinality*(e) $>$ *MIN_EXCLUSIONS*,
result ;
begin
check_in_window:
result := *in_nonexcluded_windows*(*windows*, *exclusions*, *now*) ;
print $\langle$*result*, *now*$\rangle$;
end algorithm
end algorithm;

BEGIN TRANSLATION (*chksum*(*pcal*) = "8935074*d*" $\land$ *chksum*(*tla*) = "7*c*765564")
CONSTANT *defaultInitValue*
VARIABLES *now*, *windows*, *exclusions*, *result*, *pc*

$vars \triangleq \langle now,\, windows,\, exclusions,\, result,\, pc\rangle$

$Init \triangleq$ Global variables
$\land\ now \in DATETIMES$
$\land\ windows = (\text{CHOOSE } w \in \text{SUBSET } WINDOWS : Cardinality(w) > MIN\_WINDOWS)$
$\land\ exclusions = (\text{CHOOSE } e \in \text{SUBSET } WINDOWS : Cardinality(e) > MIN\_EXCLUSIONS)$
$\land\ result = defaultInitValue$
$\land\ pc =$ "check_in_window"

$check\_in\_window \triangleq$ $\land\ pc =$ "check_in_window"
$\land\ result' = in\_nonexcluded\_windows(windows,\, exclusions,\, now)$
$\land\ PrintT(\langle result',\, now\rangle)$
$\land\ pc' =$ "Done"
$\land$ UNCHANGED $\langle now,\, windows,\, exclusions\rangle$

Allow infinite stuttering to prevent deadlock on termination.
$Terminating \triangleq pc =$ "Done" $\land$ UNCHANGED *vars*

$Next \triangleq check\_in\_window$
$\lor\ Terminating$

$Spec \triangleq Init \land \Box[Next]_{vars}$

$Termination \triangleq \Diamond(pc =$ "Done")

END TRANSLATION

\ * Modification History
\ * Last modified *Wed Jun* 16 14:01:40 *CDT* 2021 by *vputz*
\ * Created *Wed Jun* 16 10:07:32 *CDT* 2021 by *vputz*