$\overline{}$ MODULE $Directory$ $\overline{}$

EXTENDS Memory, $TLAPS$
EXTENDS $TLAPS$, $Naturals$, $FiniteSets$, $Sequences$

VARIABLES $directory$,   the current directory
    has_been_tested,   IF test has been performed THEN TRUE ELSE FALSE
    is_proved,   IF proved that directory is OK THEN TRUE ELSE FALSE
    is_error IF (raise exception OR excepetion already raised
           THEN TRUE ELSE FALSE.

Type specifies the criteria our input directory must meet:
$Type \triangleq [$
    directory has a preferences subdirectory directory/preferences.
    $preferences : [$
        directory/preferences MUST exist AND denote a directory.
      $is\_dir : \{$TRUE, FALSE$\},$
      directory/preferences MUST enclose a preferences $JSON$ file
    $preferences.JSON.$
        $has\_file : \{$TRUE, FALSE$\}$
    $]$
    $TODO$: Enrich the spec with the below constraints:
  ,input: $[$
    $is\_dir:\{$TRUE, FALSE$\}, has\_file: \{$TRUE, FALSE$\}$
  $]$
$]$

Convenience operator; see definition of $Type$.
Variables a, b must be boolean.
$Directory(a, b) \triangleq$
    $\wedge directory.preferences.is\_dir \quad = a$
    $\wedge directory.preferences.has\_file = b$

Initial condition.
$Init \triangleq$
    $\wedge directory \in Type$
    $\wedge Directory($FALSE, FALSE$)$
    $\wedge has\_been\_tested =$ FALSE
    $\wedge is\_proved \qquad =$ FALSE
    $\wedge is\_error \qquad =$ FALSE

Catch the exception "no such directory "preferences".
$NextSystemIsDirRAISE\_EXCEPTION \triangleq$
    $\wedge directory \in Type$

$\quad \land \; Directory(\text{FALSE, FALSE})$
$\quad \land \; has\_been\_tested = \text{TRUE}$
$\quad \land \; is\_proved \qquad = \text{FALSE}$
$\quad \land \; is\_error \qquad\; = \text{FALSE}$
$\quad \land \; \text{UNCHANGED } directory$
$\quad \land \; \text{UNCHANGED } has\_been\_tested$
$\quad \land \; \text{UNCHANGED } is\_proved$
$\quad \land \; is\_error' = \text{TRUE}$

Raises the above exception.
$isDirERROR \;\triangleq$
$\quad \land \; directory \in Type$
$\quad \land \; Directory(\text{FALSE, FALSE})$
$\quad \land \; has\_been\_tested = \text{TRUE}$
$\quad \land \; is\_proved \qquad = \text{FALSE}$
$\quad \land \; is\_error \qquad\;\; = \text{TRUE}$
$\quad \land \; \text{UNCHANGED } directory$
$\quad \land \; \text{UNCHANGED } has\_been\_tested$
$\quad \land \; \text{UNCHANGED } is\_proved$
$\quad \land \; \text{UNCHANGED } is\_error$

State: $Directory$ exists and is OK.
$isDirSUCCESS \;\triangleq$
$\quad \land \; directory \in Type$
$\quad \land \; Directory(\text{TRUE, FALSE})$
$\quad \land \; has\_been\_tested = \text{TRUE}$
$\quad \land \; is\_proved \qquad = \text{TRUE}$
$\quad \land \; is\_error \qquad\;\; = \text{FALSE}$
$\quad \land \; \text{UNCHANGED } directory$
$\quad \land \; \text{UNCHANGED } has\_been\_tested$
$\quad \land \; \text{UNCHANGED } is\_proved$
$\quad \land \; \text{UNCHANGED } is\_error$

Be prepared to perform the is-a-directory check:
$NextSystemCHECKisDir \;\triangleq$
$\quad \land \; Init$
$\quad \land \; directory' \in Type$

Such text is actually a success.
$NextSystemIsDirSUCCESS \;\triangleq$
$\quad \land \; NextSystemCHECKisDir$
$\quad \land \; Directory(\text{TRUE, FALSE})'$
$\quad \land \; has\_been\_tested' = \text{TRUE}$
$\quad \land \; is\_proved' \qquad = \text{TRUE}$
$\quad \land \; is\_error' \qquad\;\; = \text{FALSE}$

Such text leads to the "no such directory "preferences" exception.
$NextSystemIsDirFAILURE \triangleq$
    $\wedge\ NextSystemCHECKisDir$
    $\wedge\ Directory(\text{FALSE},\ \text{FALSE})'$
    $\wedge\ has\_been\_tested' = \text{TRUE}$
    $\wedge\ is\_proved' \qquad = \text{FALSE}$
    $\wedge\ is\_error' \qquad\ = \text{FALSE}$

The *Next* action combines the above *Next* subactions:
$Next \triangleq$
    $\vee\ NextSystemIsDirSUCCESS$
    $\vee\ NextSystemIsDirFAILURE$
    $\vee\ NextSystemIsDirRAISE\_EXCEPTION$
    $\vee\ isDirERROR$
    $\vee\ isDirSUCCESS$

$var \triangleq \langle directory,\ has\_been\_tested,\ is\_proved,\ is\_error \rangle$

$Spec \triangleq Init \wedge \square[Next]_{var}$

\ * Modification History
\ * Last modified *Tue Aug* 08 17:37:13 *CEST* 2023 by *gcordier*
\ * Created *Tue Aug* 08 16:52:36 *CEST* 2023 by *gcordier*