

Manuel d'utilisation

Spécification formelle complète

Ecran

Service : Screen
Observers : **const** Height : [Screen] \rightarrow int
 const Width : [Screen] \rightarrow int
 CellNature : [Screen] \times int \times int \rightarrow Cell
 pre CellNature(S,x,y) **requires** $0 \leq y < \text{Height}(S)$ **and** $0 \leq x < \text{Width}(S)$
Constructors : **init** : int \times int \rightarrow [Screen]
 pre **init**(h,w) **requires** $0 < h$ **and** $0 < w$
Operators : **Dig** : [Screen] \times int \times int \rightarrow [Screen]
 pre **Dig**(S,x,y) **requires** CellNature(S,x,y) = **PLT**
 Fill : [Screen] \times int \times int \rightarrow [Screen]
 pre **Dig**(S,x,y) **requires** CellNature(S,x,y) = **HOL**
Observations :
 [init] : Height(**init**(h,w)) = h
 Width(**init**(h,w)) = w
 forall (u,v) **in** [0;Width(S)] \times [0;Height(S)], CellNature(**init**(h,w),x,y) = **EMP**
 [Dig] : CellNature(**Dig**(S,x,y)),x,y = **HOL**
 forall (u,v) **in** [0;Width(S)] \times [0;Height(S)],
 (x \neq u **or** y \neq v) **implies** CellNature(**Dig**(S,x,y)),u,v) = CellNature(u,v)
 [Fill] : CellNature(**Fill**(S,x,y),x,y) = **PLT**
 forall (u,v) **in** [0;Width(S)] \times [0;Height(S)],
 (x \neq u **or** y \neq v) **implies** CellNature(**Fill**(S,x,y)),u,v) = CellNature(u,v)

Ecran editable

Service : EditableScreen **includes** Screen
Observers : Playable : [EditableScreen] \rightarrow bool
Operators : **SetNature** : [EditableScreen] \times int \times int \times Cell \rightarrow [EditableScreen]
 pre **SetNature**(S,x,y,C) **requires** $0 \leq y < \text{Height}(S)$ **and** $0 \leq x < \text{Width}(S)$
Observations :
 [invariant] : Playable(S) **min**
 forall (x,y) **in** [0;Width(S)] \times [0;Height(S)], CellNature(S,x,y) \neq **HOL**
 and forall x **in** [0;Width(S)], CellNature(S,x,0) = **MTL**
 [SetNature] : CellNature(**SetNature**(S,x,y,C)),x,y = C
 forall (u,v) **in** [0;Width(S)] \times [0;Height(S)],
 (x \neq u **or** y \neq v) **implies** CellNature(**SetNature**(S,x,y,C)),u,v) = CellNature(u,v)

Environnement

Service : Environment **includes** Screen
Observers : **CellContent** : $\text{int} \times \text{int} \rightarrow \text{Set}\{\text{Character} + \text{Item}\}$
 pre **CellContent**(E,x,y) **requires** $0 \leq y < \text{Height}(S)$ and $0 \leq x < \text{Width}(S)$
Constructors : **init** : EditableScreen \rightarrow Environment
Observations :
 [invariant] : **forall** (x,y) in $[0;\text{Width}(E)] \times [0;\text{Height}(E)]$,
 forall Character c1, c2 in **CellContent**(E,x,y)², c1 = c2
 forall (x,y) in $[0;\text{Width}(E)] \times [0;\text{Height}(E)]$,
 CellNature(E,x,y) in {MTL, PLR} **implies** **CellContent**(x,y) = \emptyset
 forall (x,y) in $[0;\text{Width}(E)] \times [0;\text{Height}(E)]$,
 exists Treasure t in **CellContent**(E,x,y)
 implies (**CellNature**(E,x,y) = EMP and **CellNature**(E,x,y-1) in {PLT, MTL})
 [init] : **forall** (x,y) in $[0;\text{Width}(E)] \times [0;\text{Height}(E)]$,
 CellNature(init(S),x,y) = EditableScreen : :**CellNature**(S,x,y)

Personnage

Service : Character
Observers : **const** Envi : [Character] \rightarrow Environment
Hgt : [Character] \rightarrow int
Wdt : [Character] \rightarrow int
Operators : **init** : Screen \times int \times int \rightarrow [Character]
pre init(S,x,y) **requires** Environment : :CellNature(S,x,y) = **EMP**
GoLeft : [Character] \rightarrow [Character]
GoRight : [Character] \rightarrow [Character]
GoUp : [Character] \rightarrow [Character]
GoDown : [Character] \rightarrow [Character]
Observations :
[**invariant**] : Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)) **in** {**EMP, HOL, LAD, HDR**}
exists Character **x** **in** Environment : :CellContent(Envi(C),Wdt(C),Hgt(C)) **implies** **x** = C
[GoLeft] : Hgt(GoLeft(C)) = Hgt(C)
Wdt(C) = 0 **implies** Wdt(GoLeft(C)) = Wdt(C)
Environment : :CellNature(Envi(C),Wdt(C)-1,Hgt(C)) **in** {**MTL, PLT**}
implies Wdt(GoLeft(C)) = Wdt(C)
Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)) **not in** {**LAD, HDR**}
and Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)-1) **not in** {**PLT, MTL, LAD**}
and not exists Character c **in** Environment : :CellContent(Envi(C),Wdt(C),Hgt(C)-1)
implies Wdt(GoLeft(C)) = Wdt(C)
exists Character c **in** Environment : :CellContent(Envi(C),Wdt(C)-1,Hgt(C))
implies Wdt(GoLeft(C)) = Wdt(C)
(Wdt(C) \neq 0) **and** Environment : :CellNature(Envi(C),Wdt(C)-1,Hgt(C)) **not in** {**MTL, PLT**}
and (Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)) **in** {**LAD, HDR**}
or Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)-1) **in** {**PLT, MTL, LAD**}
or exists Character c **in** Environment : :CellContent(Envi(C),Wdt(C),Hgt(C)-1))
and not (exists Character c **in** Environment : :CellContent(Envi(C),Wdt(C)-1,Hgt(C)))
implies Wdt(GoLeft(C)) = Wdt(C)-1
[GoRight] : Hgt(GoRight(C)) = Hgt(C)
Wdt(C) = Environment : :Width(Envi(C)) **implies** Wdt(GoRight(C)) = Wdt(C)
Environment : :CellNature(Envi(C),Wdt(C)+1,Hgt(C)) **in** {**MTL, PLT**}
implies Wdt(GoRight(C)) = Wdt(C)
Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)) **not in** {**LAD, HDR**}
and Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)-1) **not in** {**PLT, MTL, LAD**}
and not exists Character c **in** Environment : :CellContent(Envi(C),Wdt(C),Hgt(C)-1)
implies Wdt(GoRight(C)) = Wdt(C)
exists Character c **in** Environment : :CellContent(Envi(C),Wdt(C)+1,Hgt(C))
implies Wdt(GoRight(C)) = Wdt(C)
(Wdt(C) \neq Environment : :Height(Envi(C))
and Environment : :CellNature(Envi(C),Wdt(C)+1,Hgt(C)) **not in** {**MTL, PLT**}
and (Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)) **in** {**LAD, HDR**}
or Environment : :CellNature(Envi(C),Wdt(C),Hgt(C)-1) **in** {**PLT, MTL, LAD**}
or exists Character c **in** Environment : :CellContent(Envi(C),Wdt(C),Hgt(C)-1))
and not (exists Character c **in** Environment : :CellContent(Envi(C),Wdt(C)+1,Hgt(C)))
implies Wdt(GoRight(C)) = Wdt(C)+1

[GoUp] : $\text{Wdt}(\text{GoUp}(C)) = \text{Wdt}(C)$
 $\text{Hgt}(C) = \text{Environment} : : \text{Height}(\text{Envi}(C))$ **implies** $\text{Hgt}(\text{GoUp}(C)) = \text{Hgt}(C)$
 $\text{Environment} : : \text{CellNature}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)) \neq \mathbf{LAD}$ **implies** $\text{Hgt}(\text{GoUp}(C)) = \text{Hgt}(C)$
 $\text{Environment} : : \text{CellNature}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)+1) \neq \mathbf{LAD}$
 implies $\text{Hgt}(\text{GoUp}(C)) = \text{Hgt}(C)$
 exists Character c **in** $\text{Environment} : : \text{CellContent}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)+1)$ **implies**
 $\text{Hgt}(\text{GoUp}(C)) = \text{Hgt}(C)$
 $\text{Hgt}(C) \neq \text{Environment} : : \text{Height}(\text{Envi}(C))$
 and $\text{Environment} : : \text{CellNature}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)+1) = \mathbf{LAD}$
 and $\text{Environment} : : \text{CellNature}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)) = \mathbf{LAD}$
 and not exists Character c **in** $\text{Environment} : : \text{CellNature}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)+1)$
 implies $\text{Wdt}(\text{GoUp}(C)) = \text{Hgt}(C)+1$
[GoDown] : $\text{Wdt}(\text{GoDown}(C)) = \text{Wdt}(C)$
 $\text{Hgt}(C) = 0$ **implies**
 $\text{Hgt}(\text{GoDown}(C)) = \text{Hgt}(C)$
 $\text{Environment} : : \text{CellNature}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)-1)$ in **{ MTL,PLT }**
 implies $\text{Hgt}(\text{GoDown}(C)) = \text{Hgt}(C)$
 exists Character c **in** $\text{Environment} : : \text{CellContent}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)-1)$
 implies $\text{Hgt}(\text{GoDown}(C)) = \text{Hgt}(C)$
 $\text{Hgt}(C) \neq 0$
 and $\text{Environment} : : \text{CellNature}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)-1)$ in **{ MTL,PLT }**
 and not exists Character c **not in** $\text{Environment} : : \text{CellNature}(\text{Envi}(C), \text{Wdt}(C), \text{Hgt}(C)-1)$
 implies $\text{Wdt}(\text{GoDown}(C)) = \text{Hgt}(C)-1$

Garde

Service : Guard **includes** Character
Observers : **const** Id : [Guard] \rightarrow int
 Behaviour : [Guard] \rightarrow Move
 Target : [Guard] \rightarrow Character
 TimeInHole : [Guard] \rightarrow int
Operators : **init** : Screen \times int \times int \times Player \rightarrow [Guard]
 pre **init**(s,x,y,p) **requires** Environment : :CellNature(S,x,y) = **EMP** **and** p $\neq \emptyset$
 ClimbLeft : [Guard] \rightarrow [Guard]
 pre ClimbLeft(G) **requires** Environment : :CellNature(Envi(G),Hgt(G),Wdt(G)) = **HOL**
 ClimbRight : [Guard] \rightarrow [Guard]
 pre ClimbRight(G) **requires** Environment : :CellNature(Envi(G),Hgt(G),Wdt(G)) = **HOL**
 Step : [Guard] \rightarrow [Guard]
Observations :
 [invariant] : Environment : :CellNature(Envi(G),Wdt(G),Hgt(G)) = **LAD**
 and Hgt(G) < Character : :Hgt(Target(G))
 and (Environment : :CellNature(Envi(G),Wdt(G),Hgt(G),Hgt(G)-1) **not in** {**PLT**, **MTL**}
 or exists Character c **in** Environment : :CellContent(Envi(G),Wdt(G),Hgt(G)-1)
 implies Environment : :Hgt(Target(G)) - Hgt(G) <
 |Environment : :Wdt(Target(G)) - Wdt(G)|)
 implies Behaviour(G) = **Up**
 Environment : :CellNature(Envi(G),Wdt(G),Hgt(G)) = **LAD**
 and Hgt(G) > Character : :Hgt(Target(G))
 and (Environment : :CellNature(Envi(G),Wdt(G),Hgt(G),Hgt(G)-1) **not in** {**PLT**, **MTL**}
 or exists Character c **in** Environment : :CellContent(Envi(G),Wdt(G),Hgt(G)-1)
 implies Environment : :Hgt(Target(G)) - Hgt(G) >
 |Environment : :Wdt(Target(G)) - Wdt(G)|)
 implies Behaviour(G) = **Down**
 [init] : Target(G) = p
 [ClimbLeft] : Wdt(G) = 0 **implies** Wdt(ClimbLeft(G)) = Wdt(G) **and** Hgt(ClimbLeft(G)) = Hgt(G)
 Environment : :CellNature(Envi(G),Wdt(G)-1,Hgt(G)+1) **in** {**MTL**, **PLT**}
 implies Wdt(ClimbLeft(G)) = Wdt(G) **and** Hgt(ClimbLeft(G)) = Hgt(G)
 exists Character c **in** Environment : :CellContent(Envi(G),Wdt(G)-1,Hgt(G)+1)
 implies Wdt(ClimbLeft(G)) = Wdt(G) **and** Hgt(ClimbLeft(G)) = Hgt(G)
 Wdt(G) \neq 0 **and** Environment : :CellNature(Envi(G),Wdt(G)-1,Hgt(G)+1) **not in** {**MTL**, **PLT**}
 and not exists Character c **in** Environment : :CellContent(Envi(G),Wdt(G)-1,Hgt(G)+1)
 implies Wdt(ClimbLeft(G)) = Wdt(G)-1 **and** Hgt(ClimbLeft(G)) = Hgt(G)+1
 [ClimbRight] : Wdt(G) = Environment : :Width(G)
 implies Wdt(ClimbRight(G)) = Wdt(G) **and** Hgt(ClimbRight(G)) = Hgt(G)
 Environment : :CellNature(Envi(G),Wdt(G)+1,Hgt(G)+1) **in** {**MTL**, **PLT**}
 implies Wdt(ClimbRight(G)) = Wdt(G) **and** Hgt(ClimbRight(G)) = Hgt(G)
 exists Character c **in** Environment : :CellContent(Envi(G),Wdt(G)+1,Hgt(G)+1)
 implies Wdt(ClimbRight(G)) = Wdt(G) **and** Hgt(ClimbLeft(G)) = Hgt(G)
 Wdt(G) \neq Environment : :Width(G) **and**
 Environment : :CellNature(Envi(G),Wdt(G)-1,Hgt(G)+1) **not in** {**MTL**, **PLT**}
 and not exists Character c **in** Environment : :CellContent(Envi(G),Wdt(G)+1,Hgt(G)+1)
 implies Wdt(ClimbRight(G)) = Wdt(G)+1 **and** Hgt(ClimbLeft(G)) = Hgt(G)+1

[Step] : **WillFall(G) defined by** (Environment : :CellNature(Envi(G),Wdt(G),Hgt(G)-1) **in** {**HOL**, **EMP** }
and not exists Character c **in** Environment : :CellContent(Envi(G),Wdt(G),Hgt(G)-1)
and Environment : :CellNature(Envi(G),Wdt(G),Hgt(G)) **not in** {**LAD**, **HDR** })
WillIncTime(G) defined by (Environment : :CellNature(Envi(G),Wdt(G),Hgt(G)) = **HOL** **and**
TimeInHole(G) < 5)
WillLeft(G) defined by (Environment : :CellNature(Envi(G),Wdt(G),Hgt(G)) = **HOL** **and**
TimeInHole(G) = 5 **and** Behavior(G) = Left)
WillRight(G) defined by (Environment : :CellNature(Envi(G),Wdt(G),Hgt(G)) = **HOL** **and**
TimeInHole(G) = 5 **and** Behavior(G) = Right)
WillNeutral(G) defined by (Environment : :CellNature(Envi(G),Wdt(G),Hgt(G)) = **HOL** **and**
TimeInHole = 5 **and** Behavior = Neutral)
WillGoLeft(G) defined by Behavior(G) = Left
WillGoRight(G) defined by Behavior(G) = Right
WillGoUp(G) defined by Behavior(G) = Up
WillGoDown(G) defined by Behavior(G) = Down
WillFall(G) implies GoDown(G)
WillTimeInc(G) implies TimeInHole(Step(G)) = TimeInHole(G) +1
WillLeft(G) implies ClimbLeft (G)
WillRight(G) implies ClimbRight (G)
WillNeutral(G) implies Hgt(Step(G)) = Hgt(G) **and** Wdt(Step(G)) = Wdt(G)
WillGoLeft(G) implies GoLeft (G)
WillGoRight(G) implies GoRight (G)
WillGoUp(G) implies GoUp (G)
WillGoDown(G) implies GoDown (G)

Joueur

Service : Player **includes** Character
Observers : Engine : [Player] \rightarrow Engine
Operators : init : Screen \times int \times int \times Engine \rightarrow [Player]
 pre init(S,x,y,e) **requires** Environment : :CellNature(S,x,y) = **EMP** and $e \neq \emptyset$
 Step : [Player] \rightarrow [Player]
Observations :
 [init] : **Engine(P)** = e
 [Step] : **WillFall(P)** **defined by** (Environment : :CellNature(Envi(P),Wdt(P),Hgt(P)-1)
 in {**HOL**, **EMP** }
 and not exists Character c **in** Environment : :CellContent(Envi(G),Wdt(P),Hgt(P)-1)
 and Environment : :CellNature(Envi(P),Wdt(P),Hgt(P)) **not in** {**LAD**, **HDR** })
WillGoLeft(P) **defined by** Engine : :NextCommand(Engine(P)) = Left
WillGoRight(P) **defined by** Engine : :NextCommand(Engine(P)) = Right
WillGoUp(P) **defined by** Engine : :NextCommand(Engine(P)) = Up
WillGoDown(P) **defined by** Engine : :NextCommand(Engine(P)) = Down
WillDigL(P) **defined by** Engine : :NextCommand(Engine(P)) = DigL
 and (Environment : :CellNature(Envi(P),Wdt(P),Hgt(P)-1) **in** {**PLT**, **MTL**, **LAD**}
 or exists Character c **in** Environment : :CellContent(Envi(P),Wdt(P),Hgt(P)-1))
 and Environment : :CellNature(Envi(P),Wdt(P)-1,Hgt(P)) **not in** {**PLT**, **MTL**}
 and Environment : :CellNature(Envi(P),Wdt(P)-1,Hgt(P)-1) = **PLT**
WillDigR(P) **defined by** Engine : :NextCommand(Engine(P)) = DigR
 and (Environment : :CellNature(Envi(P),Wdt(P),Hgt(P)-1) **in** {**PLT**, **MTL**, **LAD**}
 or exists Character c **in** Environment : :CellContent(Envi(P),Wdt(P),Hgt(P)-1))
 and Environment : :CellNature(Envi(P),Wdt(P)+1,Hgt(P)) **not in** {**PLT**, **MTL**}
 and Environment : :CellNature(Envi(P),Wdt(P)+1,Hgt(P)+1) = **PLT**
WillGoLeft(P) **implies** GoLeft (P)
WillGoRight(P) **implies** GoRight (P)
WillGoUp(P) **implies** GoUp (P)
WillGoDown(P) **implies** GoDown (P)
WillDigL(P) **implies** Environment : :Dig(Envi(P),Wdt(P)-1,Hgt(P)-1)
 and Environment : :CellNature(Envi(P),Wdt(P)-1,Hgt(P)-1) = **HOL**
WillDigR(P) **implies** Environment : :Dig(Envi(P),Wdt(P)+1,Hgt(P)+1)
 and Environment : :CellNature(Envi(P),Wdt(P)+1,Hgt(P)+1) = **HOL**

Engine

Service : Engine

Observers : **Environment** : [Engine] → Environment
 Player : [Engine] → Player
 Guards : [Engine] → Set {Guard}
 Treasure : [Engine] → Set {Treasure}
 Status : [Engine] → Status
 NextCommand : [Engine] → NextCommand
 Holes : [Engine] → Triplet {int × int × int }

Operators : **init** : EditableScreen × int × int × Pair {int × int } × Pair {int × int } → [Player]
 pre init(S,x,y,G,T) requires Environment : :CellNature(S,x,y) = **EMP**
 Step : [Engine] → [Engine]

Observations :
 [init] : **Engine(P)** = e
 [Step] : **WillFall(P) defined by** (Environment : :CellNature(Envi(P),Wdt(P),Hgt(P)-1)
 in {**HOL**, **EMP** }
 and not exists Character c **in** Environment : :CellContent(Envi(G),Wdt(P),Hgt(P)-1)
 and Environment : :CellNature(Envi(P),Wdt(P),Hgt(P)) **not in** {**LAD**, **HDR** })
 WillGoLeft(P) defined by Engine : :NextCommand(Engine(P)) = Left
 WillGoRight(P) defined by Engine : :NextCommand(Engine(P)) = Right
 WillGoUp(P) defined by Engine : :NextCommand(Engine(P)) = Up
 WillGoDown(P) defined by Engine : :NextCommand(Engine(P)) = Down
 WillDigL(P) defined by Engine : :NextCommand(Engine(P)) = DigL
 and (Environment : :CellNature(Envi(P),Wdt(P),Hgt(P)-1) **in** {**PLT**, **MTL**, **LAD**}
 or exists Character c **in** Environment : :CellContent(Envi(P),Wdt(P),Hgt(P)-1))
 and Environment : :CellNature(Envi(P),Wdt(P)-1,Hgt(P)) **not in** {**PLT**, **MTL**}
 and Environment : :CellNature(Envi(P),Wdt(P)-1,Hgt(P)-1) = **PLT**
 WillDigR(P) defined by Engine : :NextCommand(Engine(P)) = DigR
 and (Environment : :CellNature(Envi(P),Wdt(P),Hgt(P)-1) **in** {**PLT**, **MTL**, **LAD**}
 or exists Character c **in** Environment : :CellContent(Envi(P),Wdt(P),Hgt(P)-1))
 and Environment : :CellNature(Envi(P),Wdt(P)+1,Hgt(P)) **not in** {**PLT**, **MTL**}
 and Environment : :CellNature(Envi(P),Wdt(P)+1,Hgt(P)+1) = **PLT**
 WillGoLeft(P) implies GoLeft (P)
 WillGoRight(P) implies GoRight (P)
 WillGoUp(P) implies GoUp (P)
 WillGoDown(P) implies GoDown (P)
 WillDigL(P) implies Environment : :Dig(Envi(P),Wdt(P)-1,Hgt(P)-1)
 and Environment : :CellNature(Envi(P),Wdt(P)-1,Hgt(P)-1) = **HOL**
 WillDigR(P) implies Environment : :Dig(Envi(P),Wdt(P)+1,Hgt(P)+1)
 and Environment : :CellNature(Envi(P),Wdt(P)+1,Hgt(P)+1) = **HOL**

Description formelle des tests MBT effectués

rapport de projet