Service: Level

type: int, boolean, Nature
enum: Nature {EMPTY, DIRT, METAL}

Observateurs

getHeight: [Level] → int
getWidth: [Level] → int
isEditing: [Level] → boolean
getNature: [Level] x int x int → Nature
    pre getNature(L, x, y) require x ∈ [0; getHeight(L)[∧ y ∈ [0; getWidth(L)[
getEntrance: [Level] → Point
getExit: [Level] → Point
isEntrance: [Level] x int x int → boolean
    pre isEntrance(L, x, y) require x ∈ [0; getHeight(L)[∧ y ∈ [0; getWidth(L)[
isExit: [Level] x int x int → boolean
    pre isEntrance(L, x, y) require x ∈ [0; getHeight(L)[∧ y ∈ [0; getWidth(L)[

Constructeurs

init: int x int → [Level]
    pre init(x, y) require x > 0 ∧ y > 0

Operateurs

setNature: [Level] x int x int x Nature → [Level]
    pre setNature(L, x, y, N) require x ∈ [0; getHeight(L)[∧ y ∈ [0; getWidth(L)[
        ∧ isEditing(L)

setEntrance: [Level] x int x int → [Level]
    pre setEntrance(L, x, y) require getNature(L, x, y) = EMPTY
        ∧ getNature(L, x, y-1) = EMPTY
        ∧ getNature(L, x, y+1) = EMPTY

setExit: [Level] x int x int → [Level]
    pre setExit(L, x, y) require getNature(L, x, y) = EMPTY
        ∧ getNature(L, x, y-1) = EMPTY
        ∧ getNature(L, x, y+1) = METAL

goPlay: [Level] → [Level]
    pre goPlay(L) require isEditing(L)
        ∧ ∀ j ∈ [0;getHeight(L)[, getNature(L, 0, j) = METAL
        ∧ ∀ j ∈ [0;getHeight(L)[, getNature(L, getWidth(L)-1, j) = METAL
        ∧ ∀ i ∈ [0;getWidth(L)[, getNature(L, i, 0) = METAL
        ∧ ∀ i ∈ [0;getWidth(L)[, getNature(L, i, getHeight(L)-1) = METAL

remove: [Level] x int x int → [Level]
pre remove(L, x, y) require ¬isEditing(L) ∧ getNature(L, x, y) = DIRT

build: [Level] x int x int → [Level]
pre remove(L, x, y) require ¬isEditing(L) ∧ getNature(L, x, y) = EMPTY
∧ ¬isEntrance(L, x, y) ∧ ¬isExit(L, x, y)

goEditing: [Level] → [Level]
pre goEditing(L) require ¬isEditing(L)

Observations

[init]
getHeight(init(w,h)) = h
getWidth(init(w,h)) = w
isEditing(init(w,h)) = true
∀ x ∈ [0;w[∧ ∀ y ∈ [0;h[, getNature(init(w,h)) = EMPTY

[setNature]
getNature(setNature(L, x, y, N), x, y) = N
(i, j) ≠ (x, y) ⇒ getNature(setNature(L, x, y, N), i, j) = getNature(L, x, y)

[setEntrance]
isEntrance(setEntrance(x, y), x, y) = true

[setExit]
isExit(setExit(x, y), x, y) = true

[goPlay]
isEditing(goPlay(L)) = false
∃! (x, y), isEntrance(L, x, y) ∧ getNature(L, x, y-1) = EMPTY
∧ getNature(L, x, y+1) = EMPTY
∃! (x, y), isExit(L, x, y) ∧ getNature(L, x, y-1) = EMPTY
∧ getNature(L, x, y+1) = METAL

[remove]
getNature(remove(L, x, y), x, y) = EMPTY

[build]
getNature(build(L, x, y), x, y) = DIRT

[goEditing]
isEditing(goEditing(L)) = true