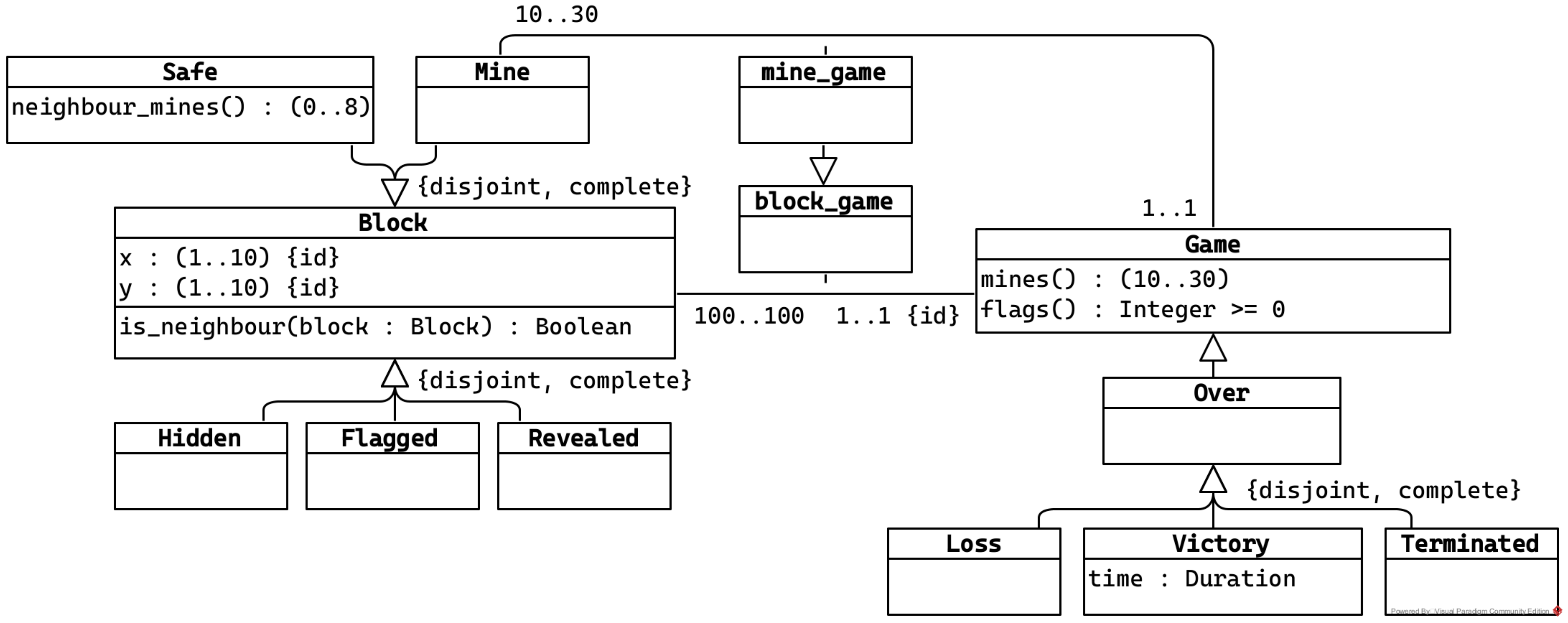
Minesweeper 💣

Table of contents

UML



10..30
Safe
neighbour_mines() : (0..8)
Mine
mine_game
block_game
{disjoint, complete}
1..1
Block
x : (1..10) {id}
y : (1..10) {id}
is_neighbour(block : Block) : Boolean
100..100
1..1 {id}
Game
mines() : (10..30)
flags() : Integer >= 0
{disjoint, complete}
Hidden
Flagged
Revealed
Over
{disjoint, complete}
Loss
Victory
time : Duration
Terminated
Powered By: Visual Paradigm Community Edition

Game class specification

```
[V.Game.at_most_one_uncovered_mine]
  ∀ game
      Game(game) ⟹
          ¬ ∃ m1, m2
              m1 ≠ m2 ∧
              mine_game(m1, game) ∧
              mine_game(m2, game) ∧
              Revealed(m1) ∧
              Revealed(m2)
```

```
[V.Game.victory_condition]
  ∀ game
      Victory(game) ⟺
          ∀ block mine_game(block, game) ⟹ Flagged(block) ∧
          ¬ ∃ block block_game(block, game) ∧ Safe(block) ∧ Flagged(block)
          ∨
          ∀ block (block_game(block, game) ∧ Safe(block) ⟹ Revealed(block))
```

```
[V.Game.loss_condition]
  ∀ game
      Loss(game) ⟺ ∃ mine mine_game(mine, game) ∧ Revealed(mine)
```

```
[V.Game.only_one_uncompleted_game]
  ∀ g1, g2
      Game(g1) ∧ Game(g2) ∧ g1 ≠ g2 ⟹
          Over(g1) ∨ Over(g2)
```

```
mines(): (10..30)
  pre-conditions
  post-conditions
    result = |{ mine | mine_game(mine, this) }|
```

```
flags(): Integer ≥ 0
  pre-conditions
  post-conditions
    result = |{ flag | block_game(flag, this) ∧ Flag(block) }|
```

Block class specification

```
is_neighbour(block: Block): Boolean
  pre-conditions
  post-conditions
    result = True ⟺ ∃ game, x, y
         block_game(this, game) ∧
         block_game(block, game) ∧
         x(this, x) ∧
         y(this, y) ∧
         (
              (x(block, x − 1) ∧ y(block, y − 1)) ∨
              (x(block, x ) ∧ y(block, y − 1)) ∨
              (x(block, x + 1) ∧ y(block, y − 1)) ∨
              (x(block, x − 1) ∧ y(block, y + 1)) ∨
              (x(block, x ) ∧ y(block, y + 1)) ∨
              (x(block, x + 1) ∧ y(block, y + 1)) ∨
              (x(block, x − 1) ∧ y(block, y)) ∨
              (x(block, x + 1) ∧ y(block, y)) ∨
         )
```

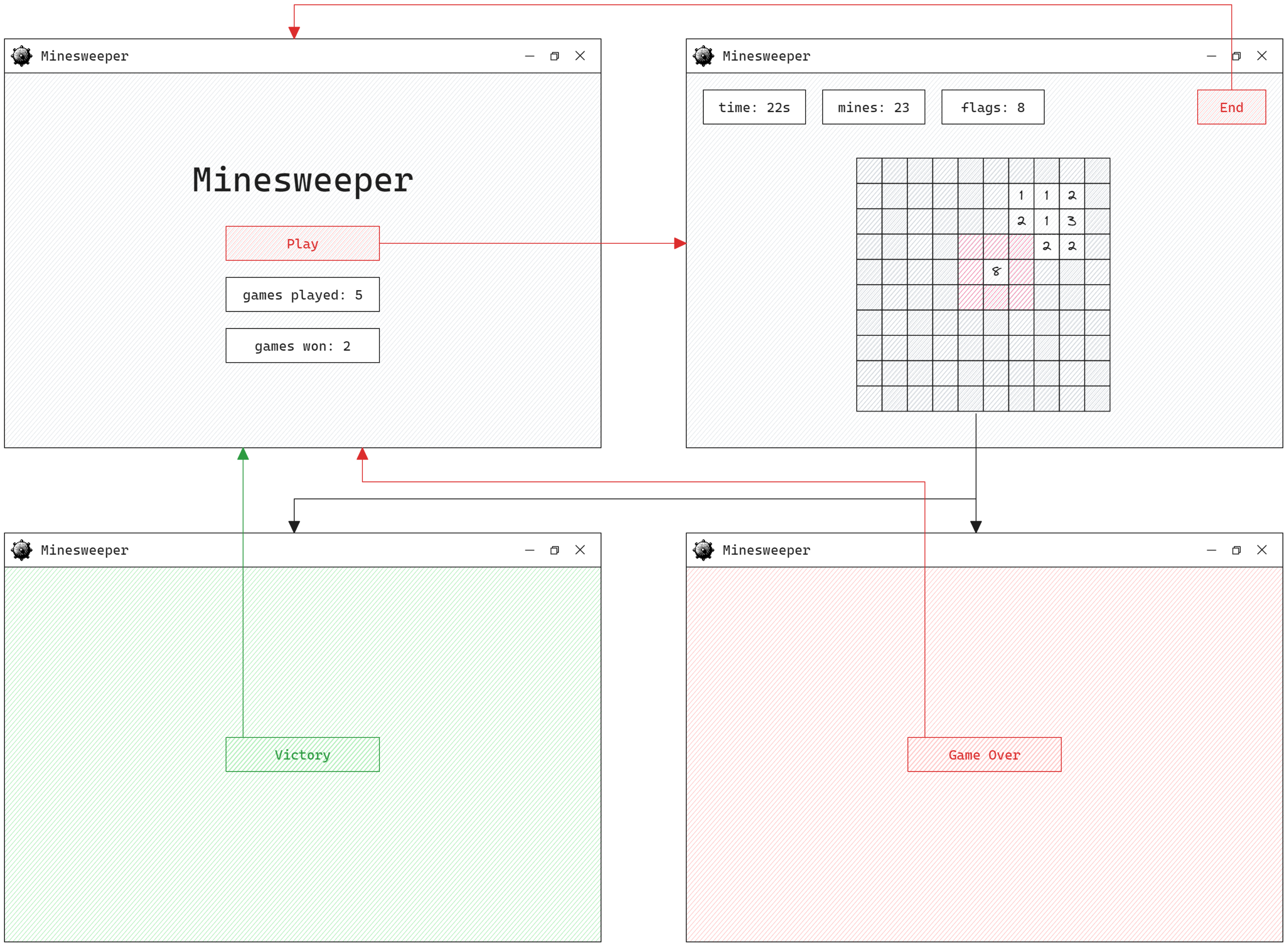
Safe class specification

```
[V.Safe.revealed_empty_block_reveals_neighbours]
  ∀ safe, game, neighbour
       block_game(safe, game) ∧
       Safe(safe) ∧
       Revealed(safe) ∧
       is_neighbour_{Block, Block, Boolean}(safe, neighbour, True) ∧
       neighbour_mines_{Safe, (0..8)}(safe, 0) ⟹
            Revealed(neighbour)

neighbour_mines(): (0..8)
  pre-conditions
  post-conditions
    result = |{ mine | ∃ game
         block_game(this, game) ∧
         mine_game(mine, game) ∧
         is_neighbour_{Block, Block, Boolean}(this, mine, True)
    }|
```

Use Case

```
start_game(): Game
  pre-conditions
    ¬ ∃ game Game(game) ∧ ¬ Over(game)

terminate_game(game: Game): Terminated
  pre-conditions
    ¬ Over(game)

reveal(block: Hidden): Revealed
  pre-conditions
    ∃ game block_game(block, game) ⟹ ¬ Over(game)

flag(block: Hidden): Flagged
  pre-conditions
    ∃ game block_game(block, game) ⟹ ¬ Over(game)

remove_flag(block: Flagged): Hidden
  pre-conditions
    ∃ game block_game(block, game) ⟹ ¬ Over(game)

games_played(): Integer ≥ 0
  pre-conditions
  post-conditions
    result = |{ game | Game(game) }|

games_won(): Integer ≥ 0
  pre-conditions
  post-conditions
    result = |{ game | Victory(game) }|
```

Wireframe


Minesweeper
Minesweeper
Play
games played: 5
games won: 2
Minesweeper
time: 22s
mines: 23
flags: 8
End
1 1 2
2 1 3
2 2
8
Minesweeper
Victory
Minesweeper
Game Over