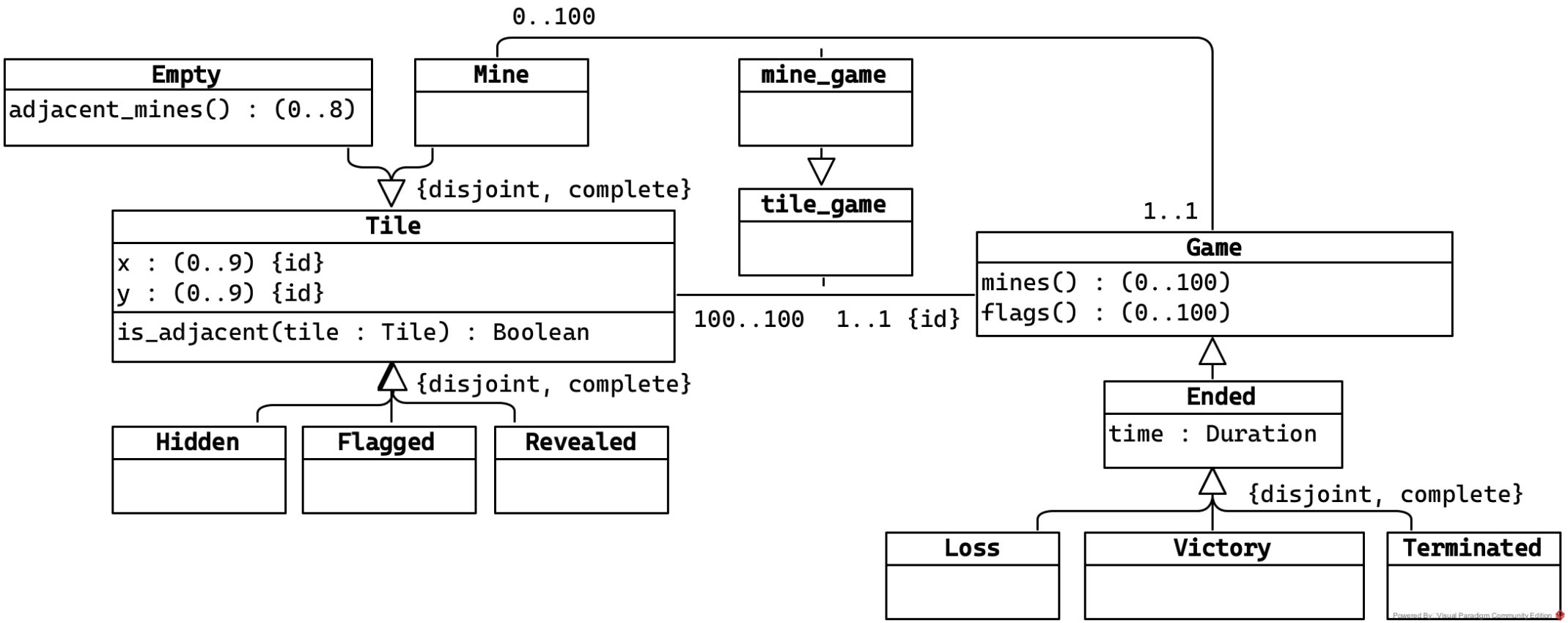
Minesweeper 💣

Table of contents

UML


0..100
Empty
adjacent_mines() : (0..8)
Mine
mine_game
tile_game
{disjoint, complete}
Tile
x : (0..9) {id}
y : (0..9) {id}
is_adjacent(tile : Tile) : Boolean
100..100
1..1 {id}
1..1
Game
mines() : (0..100)
flags() : (0..100)
{disjoint, complete}
Hidden
Flagged
Revealed
Ended
time : Duration
{disjoint, complete}
Loss
Victory
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) ⟺
        ∀ tile mine_game(tile, game) ⟹ Flagged(tile) ∧
        ¬ ∃ tile tile_game(tile, game) ∧ Empty(tile) ∧ Flagged(tile)
        ∨
        ∀ tile (tile_game(tile, game) ∧ Empty(tile) ⟹ Revealed(tile))
```

[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 ⟹
        Ended(g1) ∨ Over(g2)
```

mines(): (0..100)

pre-conditions

post-conditions

```
result = |{ mine | mine_game(mine, this) }|
```

flags(): (0..100)

pre-conditions

post-conditions

```
result = |{ flag | tile_game(flag, this) ∧ Flag(tile) }|
```

Tile class specification

```
is_adjacent(tile: Tile): Boolean
```

pre-conditions

post-conditions

```
result = True ⟺ ∃ game, x, y
    tile_game(this, game) ∧
    tile_game(tile, game) ∧
    x(this, x) ∧
    y(this, y) ∧
    (
        (x(tile, x - 1) ∧ y(tile, y - 1)) ∨
        (x(tile, x ) ∧ y(tile, y - 1)) ∨
        (x(tile, x + 1) ∧ y(tile, y - 1)) ∨
        (x(tile, x - 1) ∧ y(tile, y + 1)) ∨
        (x(tile, x ) ∧ y(tile, y + 1)) ∨
        (x(tile, x + 1) ∧ y(tile, y + 1)) ∨
        (x(tile, x - 1) ∧ y(tile, y)) ∨
        (x(tile, x + 1) ∧ y(tile, y)) ∨
    )
```

Empty class specification

```
[V.Empty.revealed_empty_tile_reveals_adjacents]
```

```
∀ safe, game, adjacent
    tile_game(safe, game) ∧
    Empty(safe) ∧
    Revealed(safe) ∧
    is_adjacent_Tile, Tile, Boolean(safe, adjacent, True) ∧
    adjacent_mines_Empty, (0..8)(safe, 0) ⟹
        Revealed(adjacent)
```

```
adjacent_mines(): (0..8)
```

pre-conditions

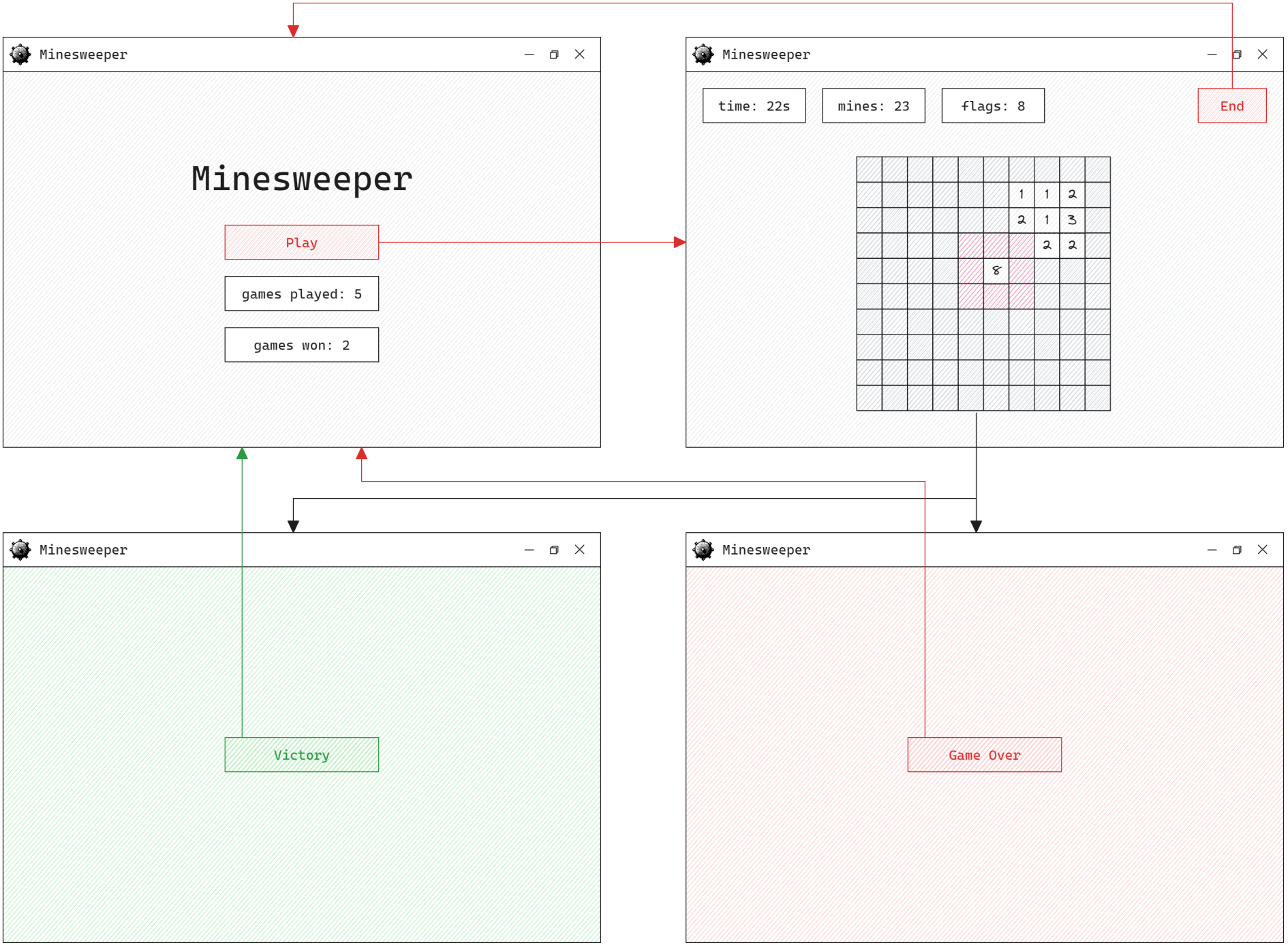
post-conditions

```
result = |{ mine | ∃ game
    tile_game(this, game) ∧
    mine_game(mine, game) ∧
    is_adjacent_Tile, Tile, Boolean(this, mine, True)
}|
```

Use Case

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

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

reveal(tile: Hidden): Revealed
  pre-conditions
    ∃ game tile_game(tile, game) ⟹ ¬ Ended(game)

flag(tile: Hidden): Flagged
  pre-conditions
    ∃ game tile_game(tile, game) ⟹ ¬ Ended(game)

remove_flag(tile: Flagged): Hidden
  pre-conditions
    ∃ game tile_game(tile, game) ⟹ ¬ Ended(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
Minesweeper
Victory
Minesweeper
Game Over