Learning Object-Oriented Programming, Design and TDD with Pharo

Stéphane Ducasse

March 11, 2019

Layout and typography based on the sbabook LaTeX class by Damien Pollet.

Contents

Illustrations

CHAPTER 1

Snake and ladder solutions

1.1 A first real test

```
SLGame >> tileNumber: aNumber

  tiles := Array new: aNumber.
  1 to: aNumber do: [ :i |
    tiles at: i put: (SLTile new position: i) ]
```

```
SLGame >> tileNumber

  ^ tiles size
```

1.2 Accessing on tile

```
SLGame >> tileAt: aNumber

  ^ tiles at: aNumber
```

1.3 Adding players

```
SLPlayer >> name: aString
  name := aString
```

```
Object subclass: #SLTile
  instanceVariableNames: 'position players'
  classVariableNames: ''
  package: 'SnakesAndLadders'
```

```
SLGame >> initialize
  players := OrderedCollection new.
```

```
SLTile >> addPlayer: aPlayer
  players add: aPlayer
```

```
SLTile >> players
  ^ players
```

1.4 Displaying players

```
SLPlayer >> printOn: aStream
  aStream << '<' << name << '>'
```

1.5 Avoid leaking implementation information

```
SLTile >> includesPlayer: aPlayer
  ^ players includes: aPlayer
```

1.6 Preparing to move players

```
SLPlayer >> position
  ^ position
```

```
SLPlayer >> position: anInteger
  position := anInteger
```

```
Object subclass: #SLPlayer
  instanceVariableNames: 'name position'
  classVariableNames: ''
  package: 'SnakesAndLadders'
```

```
SLGame >> addPlayer: aPlayer
  aPlayer position: 1.
  (tiles at: 1) addPlayer: aPlayer
```

```
SLGame >> tileFor: aPlayer atDistance: aNumber

  ^ self tileAt: (aPlayer position + aNumber)
```

1.7 Finding the tile of a player

```
SLGame >> tileOfPlayer: aSLPlayer
  ^ tiles at: aSLPlayer position
```

1.8 Moving a player

```
SLTile >> removePlayer: aPlayer
  players remove: aPlayer
```

```
SLGame >> movePlayer: aPlayer distance: anInteger
  | targetTile |
  targetTile := self tileFor: aPlayer atDistance: anInteger.
  (self tileOfPlayer: aPlayer) removePlayer: aPlayer.
  targetTile addPlayer: aPlayer.
  aPlayer position: targetTile position.
```

1.9 New printing hook

```
SLAbstractTile >> printOn: aStream

  aStream << '['.
  self printInsideOn: aStream.
  aStream << ']'
```

```
SLLadderTile >> printInsideOn: aStream

  super printInsideOn: aStream.
  aStream << '->'.
  targetTile position printOn: aStream
```

1.10 Snake and ladder declaration

```
SLGame >> setSnakeFrom: aSourcePosition to: aTargetPosition

  tiles
    at: aSourcePosition
    put: (SLSnakeTile new
        position: aSourcePosition;
        to: (tiles at: aTargetPosition) ; yourself)
```

```
SLGame >> setLadderFrom: aSourcePosition to: aTargetPosition

  tiles
    at: aSourcePosition
    put: (SLLadderTile new
        position: aSourcePosition ;
        to: (tiles at: aTargetPosition) ; yourself)
```

1.11 Active tile actions

```
SLActiveTile >> acceptPlayer: aPlayer
  targetTile acceptPlayer: aPlayer
```

1.12 Player turns and current player

```
Object subclass: #SLGame
  instanceVariableNames: 'tiles players turn'
  classVariableNames: ''
  package: 'SnakesAndLadders'
```

```
SLGame >> initialize
  players := OrderedCollection new.
  turn := 0
```

1.13 Game end

```
SLGame >> isOver

  ^ players anySatisfy: [ :each | each position = tiles size ]
```

1.14 Playing one move

```
SLGame >> canMoveToPosition: aNumber
  "We can only move if we stay within the game.
  This implies that the player should draw an exact number to land
    on the finish tile."

  ^ aNumber <= tiles size
```

Bibliography