Risolutore di puzzle Parte 1

Alberto Andeliero

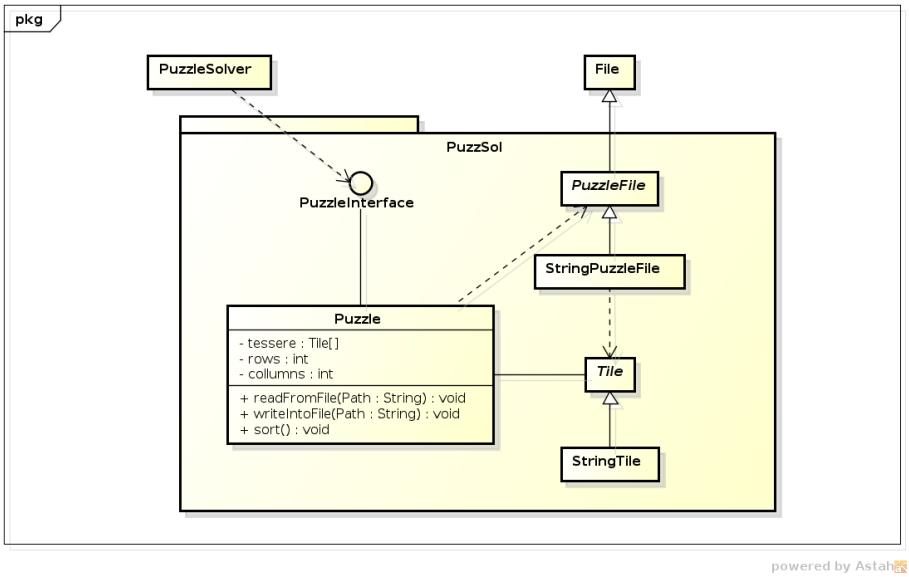
21 gennaio 2015

Programmazione concorrente e distribuita
Progetto A.A. 2014/2015

1 Scelte progettuali

PuzzleSolver come da specifica stato concepito per modellare una realt di puzzle rettangolari e di risolverne l'ordine delle tessere. Inoltre questo programma stato pensato e progettato per essere estendibile con altre tipologie di puzzle rettangolari che trattino altri oggetti anzich Stringhe al proprio interno. Quindi oltre alla classe PuzzleSolver che conterr il main dell'applicazione, stata modellata la classe Puzzle che responsabile delle operazioni quali lettura da file di input, scrittura in file di output, e fondamentale il metodo sort che riordina i tasselli del puzzle secondo quanto richiesto dalla specifica tecnica.

2 Organizzazione delle classi



Macroscopicamente possiamo vedere 3 gerarchie di classi, la gerarchia del Puzzle, la gerarchia delle tessere, ed infine la gerarchia dei PuzzleFile. La gerarchia della classe puzzle composta da PuzzleInterface e da Puzzle, si deciso di

fornire un'interfaccia generica per avere una futura estensibilit con altre tipologie di puzzle ed inoltre per la creazione di un'istanza concreta di Puzzle stato utilizzato un pattern di tipo Factory. La gerarchia delle tessere composto semplicemente da una super classe astratta Tile che implementa gi le operazioni di base e di confronto fra tessere e StringTile concretizza il fatto che la tessere costituita da un'oggetto stringa. Infine abbiamo la gerarchia dei FilePuzzle dove PuzzleFile estende la classe File e StringPuzzleFile concretizza PuzzleFile implementando i metodi astratti di input e output utilizzati dalla classe Puzzle. Tutte le gerarchie sono state incapsulate nel package puzzsol tranne PuzzleSolver per ovvi motivi.

3 Algoritmo di risoluzione

Nella classe Puzzle possiamo trovare tutta la logica di ordinamento fondamentale alla risoluzione del puzzle. Avendo il riferimento a tutte le tessere lette in input da file, e il numero di colonne e di righe precedentemente calcolato,

1. In primis cerca il primo tassello in 0-posizione verificando che per ogni tassello presente vi sia uno con nord e ovest vuoti;
2. Successivamente il flusso procede a cercare il tassello pi a est di quello precedentemente trovato, fino a completare l'intera riga;
3. Completata la riga si procede a cercare la tessera pi a sud del capo riga;
4. Si ripete dal punto 2 il flusso fino a che tutte le righe non sono complete.

Nella classe interna TileScout sono state incapsulate tutti i metodi per la ricerca utili per l'algoritmo di risoluzione.

4 Corettezza del programma