Πανεπιστήμιο Κρήτης –Τμήμα Επιστήμης Υπολογιστών

ΗΥ252– Αντικειμενοστρεφής Προγραμματισμός

Διδάσκων: Ι. Τζίτζικας

Χειμερινό Εξάμηνο 2020-2021

Εισαγωγή

Αρχικα θα φτιαξω το documentation το Μοντελο και επιτα εφοσον εχουν σχεδιαστει ολα καταληλα θα αρχισω να γραφω κωδικα.Θα γινει χρειση του MVC μοντελου οπου γινει αναλυτικει περιγραφει το πως συνδεονται ολα. Επισης θα υλοποιησουμε ενα Διαγραμμα UML.

Περιεχόμενα

- Εισαγωγή

Θα υλοποιησω το MVC(Model-View-Controller). Οπου το Model και το View θα ειναι αγνωστα μεταξυ τους δεν θα γνωριζει το ενα οτι υπαρχει το αλλο. Και το Controller το οποιο θα τα συνδεει. Στο Model θα αποθηκευονται τα Δεδομενα και η Μεθοδοι. Το View θα ειναι το Interface μας δηλαδη τα γραφικα. Αυτο το ειδους μοντελου ειναι χρεισημο διοτι μπορουμε να παρουμε κομματια του στο μελλον και να τα χρησιμοποιησουμε καπου σε καποιο αλλο προτζεκ και επισης υπαρχει μια οργανοσει που βοηθαει στο Scalling.Επιπροσθετα μπορουμε να αλλαξουμε ενα απο τα τρεια και το μοντελο να δουλευει κανονικα.

- Η Σχεδίαση και οι Κλάσεις του Πακέτου Model

Package model.tiles

- public class bag(){}

  Βαζουμε ολες τις tiles σε μια Arraylist χωρις να μας ενδιαφερουν τα ειδικα τους χαρακτηριστηκα και φτιαχνουμε μια συλλογη απο tiles. Επισης κωδικοποιουμαι τις Μεθοδους για την διαχειρισει της bag.

  Η Class αυτό μας παρέχει τις εξής μεθόδους και μεταβλητες:

  1. private ArrayList<tile> tiles; //Attribute

     Οπου θα αποθηκευσουμε τις tiles τις τσαντας.

  2. public void init_tiles(){} //Transformer(Mutative)

     Αρχηκοποιει τα περιχομαινα τις bag οχι Random.

  3.public boolean isEmpty(){} // Observer

     Επιστρεφει αν η ArrayList ειναι αδεια.

  4.public void addTile(tile i){} //Transformer(Mutative)

     Προσθετει μια tile στη ArrayList

  5.public void removeTile(){} //Transformer(Mutative)

     Αφαιρει ενα Tile απο το τελος του bag.

  6.public in size(){} //Accessor (Selector)

     Επιστρεφει το Μεγεθος τις Λιστας

  7.public void clearAll(){} //Transformer(Mutative)

     Καθαριζει την Λιστα απο τις tiles.

  8.public ArrayList<tile> return4Tiles(){}//Accessor(Selector)

     Επιστρεφει 4 tiles απο το τελος του bag.

  9.public void remove4Tiles(){} //Transformer(Mutative)

     Αφαιρει απο την Λιστα 4 tiles απο το τελος

  10.public void randomizeBag(){}//Transformer(Mutative)

     Βαζει σε τυχαια σειρα τα tiles της bag.

- abstract interface tile{}

  Θα χρεισημοποιησουμε interface γιατι θα θελουμε να μπορουμε να βαζουμε σε μια λιστα ολα τα tiles ανεξετους σε ποια κατυγορια ανοικει.

  Μεθοδοι του Interface:

- public enum tileColor

  Ειναι μια enum που περιεχει τα χρωματα των tiles τα οποια περνουν.

- public class MarbleStatues implements title{}

  Εδω θα δημιουργουμε τις Carayatids και τις Sphinxes οπου θα πρεπει να δηλωνουμε και τον τυπο τις στον constructor και οριζουμε και μια enum.

  Methods:

  1.public marbleTypes getType(){} //Accessor (Selector)

  Επιστρεφει τον type του MarbleStatues

- public enum marbleTypes{}

  Εδω ειναι οι δυο τυποι MarbleStatues που μπορουμε να ορισουμε.

- public class Landslides implements tile

  Εδω θα μπορουμε να δημιουργησουμε αντικειμενα τυπου κατολίσθησης για να καλυψουν την εισοδο.

- public class Mosaics implements tile{}

  Δημημιοργει Mosaic για το οποιο δημονουμε το χρωμμα στο constructor στα οποια τα χρωμματα τα δηλωσαμε σε μια enum tileColor.

  Methods:

  1.public tileColor getColot(){} //Accessor (Selector)

  Το οποιο επιστρεφει το χρωμμα του Mosaics.

- public class Skeletons implements tile

  Εδω μπορουμε να δημιουργισουμε Skeleton επισης να πουμε και σε ποια κατηγορια ανηκει.

  Methods:

1. public skeletonParts getPart(){} //Accessor(Selector)

Επιστρεφει τι type skeleton ειναι ποιο μερος πανω κατω μεγαλο μικρο

- public enum skeletonParts

  Ειναι μια enum που περιεχει τα διαφορα μερη σκελετων που μπορουμε να βρουμε.

- public class Amphoras implements tile{}

  Εδω θα φτιαχνουμε αντικειμενα τυπου Amphoras με το Color που εχει το

  καθενα ενα και το χρωμμα θα το δηλωνουμε ως tileColor που ειναι μια enum.

  Methods:

  1.public tileColor getColot(){} //Accessor (Selector)

  Το οποιο επιστρεφει το χρωμμα του Amphoras.

Package model.tiles

- public abstract class card{}

  Εδω εχουμε τις Methods και Attributes που εχουν ολες οι καρτες κοινες.

  Attributes:

  1. private boolean used = false;

  Methods:

  1.public abstract void useCard(); //Transformer

  Η οποια εχει διαφορετικοι υλοποιησει για καθε καρτα και πρεπει να υλοποιηθει απο την Υπο-Κλαση.

  2.public boolean isUsed(){} //Accessor

  Επιστρεφει True αν η Card εχει Χρεισημοποιηθει. Δεν ειναι abstract γιατι ειναι ειδια υλοποιησει για ολες τις καρτες.

- public class archaeologist extends card{}

  Για να μπορουμε να δημιουργισουμε αντικειμενο Archaeologist Card.

  Methods:

1.public void useCard(){}

Οπου θα γινει χρεισει τις καρτα η ικανοτητα της η οποια ειναι να Ο παίκτης να παίρνει μέχρι 2 πλακίδια από οποιαδήποτε περιοχή τοποθέτησης, εκτός από αυτή που διάλεξε νωρίτερα στη σειρά του.

- public class assistant extends card{}

Για να μπορουμε να δημιουργισουμε αντικειμενο Archaeologist Card.

Methods:

1.public void useCard(){}

Οπου θα γινει χρεισει τις καρτα η ικανοτητα της η οποια ειναι να Ο παίκτης να παίρνει 1 πλακίδιο από οποιαδήποτε περιοχή τοποθέτησης

- public class digger extends card{}

Για να μπορουμε να δημιουργισουμε αντικειμενο Digger Card.

Methods:

1.public void useCard(){}

Οπου θα γινει χρεισει τις καρτα η ικανοτητα της η οποια ειναι να Ο παίκτης να παίρνει μέχρι 2 πλακίδια από την περιοχή που διάλεξε νωρίτερα στη σειρά του.

- public class professor extends card{}

Για να μπορουμε να δημιουργισουμε αντικειμενο Professor Card.

Methods:

1.public void useCard(){}

Οπου θα γινει χρεισει τις καρτα η ικανοτητα της η οποια ειναι να Ο παίκτης να παίρνει ένα πλακίδιο από κάθε περιοχή, εκτός από αυτή που διάλεξε νωρίτερα στη σειρά του.

Package model.Player

public class Player{}

Εδω θα μπορουμε να δημιουργουμαι παιχτες για το παιχνηδη.

Methods:

1. public String getName(){} //Accessor
   Επιστρεφει το Name του Player
2. private void addCards(card c){} //Transformer
   Προσθετει Μια Card στη Λιστα με τις Cards το Player.
3. public void addTile(tile t){} //Transformer
   Προσθετει μια Tile στν Λιστ με τις Tiles που εχει μαζεψει ο Player
4. public int calculatePoints(){} //Accessor

   Υπολογιζει και επιστρεφει τους ποντους που εχει μαζεψει ο παιχτης με βασει τα Tiles.

5. public int getID(){} //Accessor

   Επιστρεφει το ID του Player

Package model.board

public class board{}

Με αυτη την κλαση θα μπορουμε να αρχικοποιουμαι το ταμπλο και να του προσθετουμαι και να αφαιρουμαι tiles απο αυτο. Επισης θα μπορουμε να ελενχουμε αν το παιχνηδη εχει τελειωσει το οποιο γινεται οταν στην περιοχη εισοδου εχουμε 16 Landslides.

Attributes:

1. private ArrayList<Mosaics>

   Εδω ειναι η περιοχη που τοποθετουμαι τα Mosaics.

2. private ArrayList<MarbleStatues>

   Εδω ειναι η περιοχη που τοποθετουμαι τα MarbleStatues.

3. private ArrayList<Amphoras>

   Εδω ειναι η περιοχη που τοποθετουμαι τα Amphoras.

4. private ArrayList<Skeletons>

   Εδω ειναι η περιοχη που τοποθετουμαι τα Skeletons.

5. pricate ArrayList<Landslides>

   Εδω ειναι η περιοχη που τοποθετουμαι τα Landslides.

Methods:

1.public boolean hasTheGameEnded() //Observer

Ελενχει αν το παιχνηδη εχει τελειωσει αν στην Εισοδο εχουμε 16 Landslides

2.public void addMosaics(Mosaics m) //Transformer

Προσθετει ενα Mosaic στην περιοχη των Mosaics

3.public void removeMosaics(Mosaics m) //Transformer

Αφαιρει ενα Mosaic απο την περιοχη Mosaic

4.public void addMarbleStatues(MarbleStatues m) //Transformer

Προσθετει ενα MarbleStatuesστην περιοχη των MarbleStatues

5.public void removeMarbleStatues(MarbleStatues m) //Transformer

Αφαιρει ενα MarbleStatue απο την περιοχη MarbleStatues

6.public void addAmphoras(Amphoras a) //Transformer

Προσθετει ενα Amphora στην περιοχη των Amphoras

7.public void removeAmphoras(Amphoras a) //Transformer

Αφαιρει ενα Amphora απο την περιοχη Amphoras

8.public void addSkeletons(Skeletons a) //Transformer

Προσθετει ενα Skeleton στην περιοχη των Skeletons

9.public void removeSkeletons(Skeletons s) //Transformer

Αφαιρει ενα Skeleton απο την περιοχη Skeletons

10.public int getNumLandslides() //Accessor

Επιστρεφει τον αριθμο των Landslides

11.public int getNumMarbleStatues() //Accessor

Επιστρεφει τον αριθμο των MarbleStatues που το τυπο ποιον των αριθμο θελουμαι μπορουμε να το επιλεξουμε απο την παραμετρο

12.public int getNumMosaics(tileColor col) //Accessor

Επιστρεφει τον αριθμο των Mosaics το χρωμα το επιλογουμε με την

13.public int getNumSkeleton(skeletonParts part) //Accessor

Επιστρεφει των αριθμο των σκελετων με το μερος που επιλεξαμε απο την παραμετρο

14.public int getAmphoras(tileColor col) //Accessor

Επιστρεδει των αριθμω των Amphoras με χρωμμα col

Package model.turn

Σε αυτη την κλαση θα μπορουμε να ελενχουμε ποιος εχει σειρα και τις δυνατοτητες που μπορει να προγματοποιησει στον γυρο του/

Attributes:

1. ArrayList<Player> players;

2.int currentID;

Methods:

1.public Player nextPlayer() //Transformer

Επιστρεφει τον παιχτη που εχει σειρα και επισεις αλλαζει το ID της κλασσεις στον παιχτη που εχει τωρα σειρα.

2.public int getID() //Accessor

Επιστρεφει το ID που παιζει τωρα

- Η Σχεδίαση και οι Κλάσεις του Πακέτου Controller

Ουσιαστηκη αυτη η κλασσι κοντρολαρη το παιχνηδη κανει ολες τις κλισης αν πατηθει κατι στο View καλη τις απαραιτιτες συναρτησεις της Model για να γινουν οι αλλαγες . Σε αυτη την κλαση Δημιουργουμαι τους Παιχτες, το ταμπλο, γυρων οπου συνδεουμαι στο τελος το View με το Model. Σε αυτη τη κλαση ελενχουμαι ποτε τελειωνει το παιχνηδη και το τερματιζουμαι και επισης εδω υπολογιζουμαι τους ποντους των παιχτων.

Methods:

1.public void next_Turn() //Transformer

Δινει στον επομενο παιχτη την σειρα του. Κανοντας αλλαγες στο View και περνοντας κυριος την πληροφορια απο το Model.Επισης σε αυτη την Μεθοδο θα βγαζουμαι 4 Πλακηδια απο την Τσαντα και θα τα τοποθετουμαι στο ταμπλο.

2.public int seeTurn() //Accessor

Επιστρεφει το ID του τωρινου παιχτη.

3.public boolean game_has_finished() //Observer

Επιστρεφει True αν το παιχνηδη εχει τελειωσει αν στην περιοχη εισοδου εχουμε 16 Landslides.

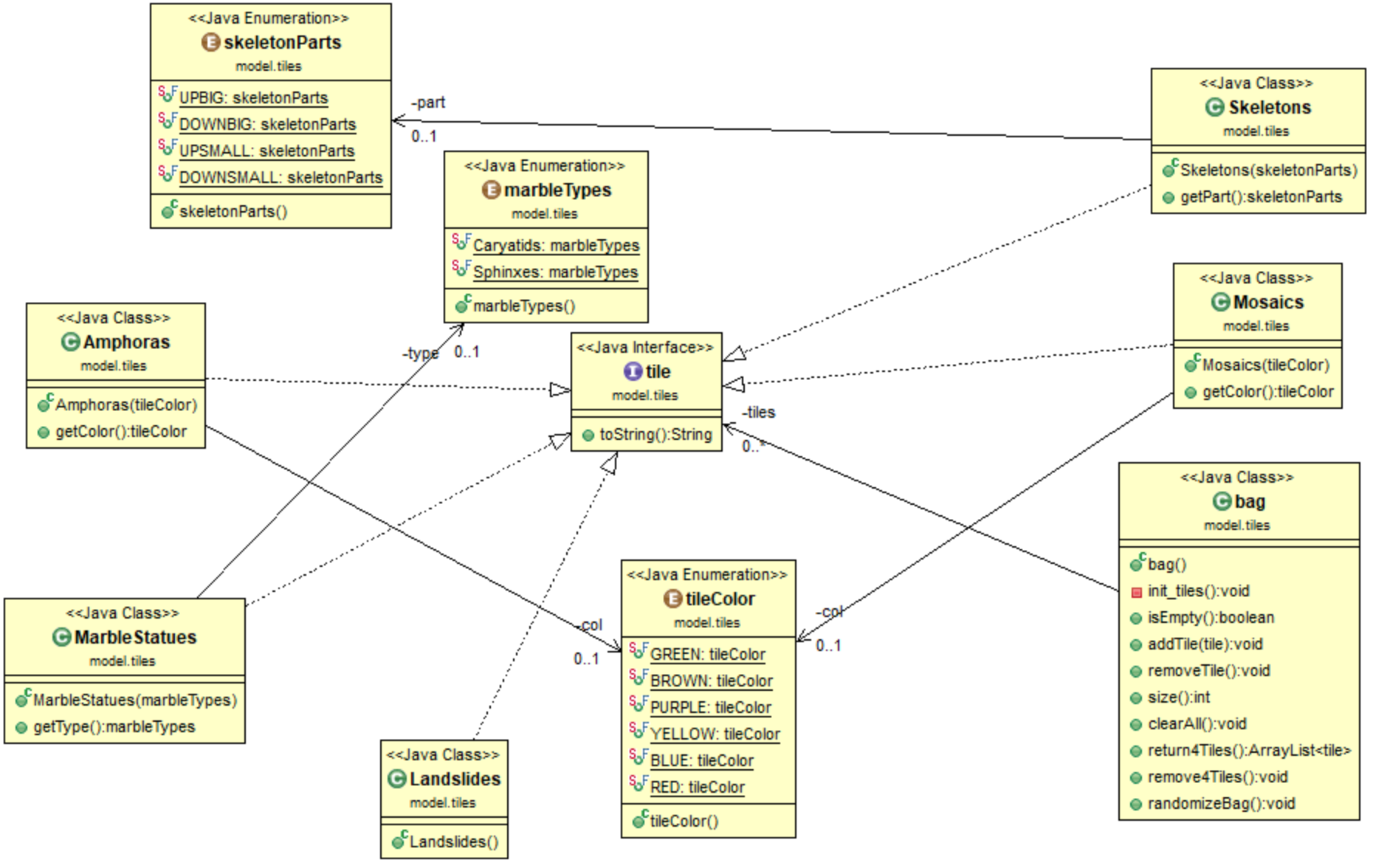
4.public Player getWinner() //Accessor

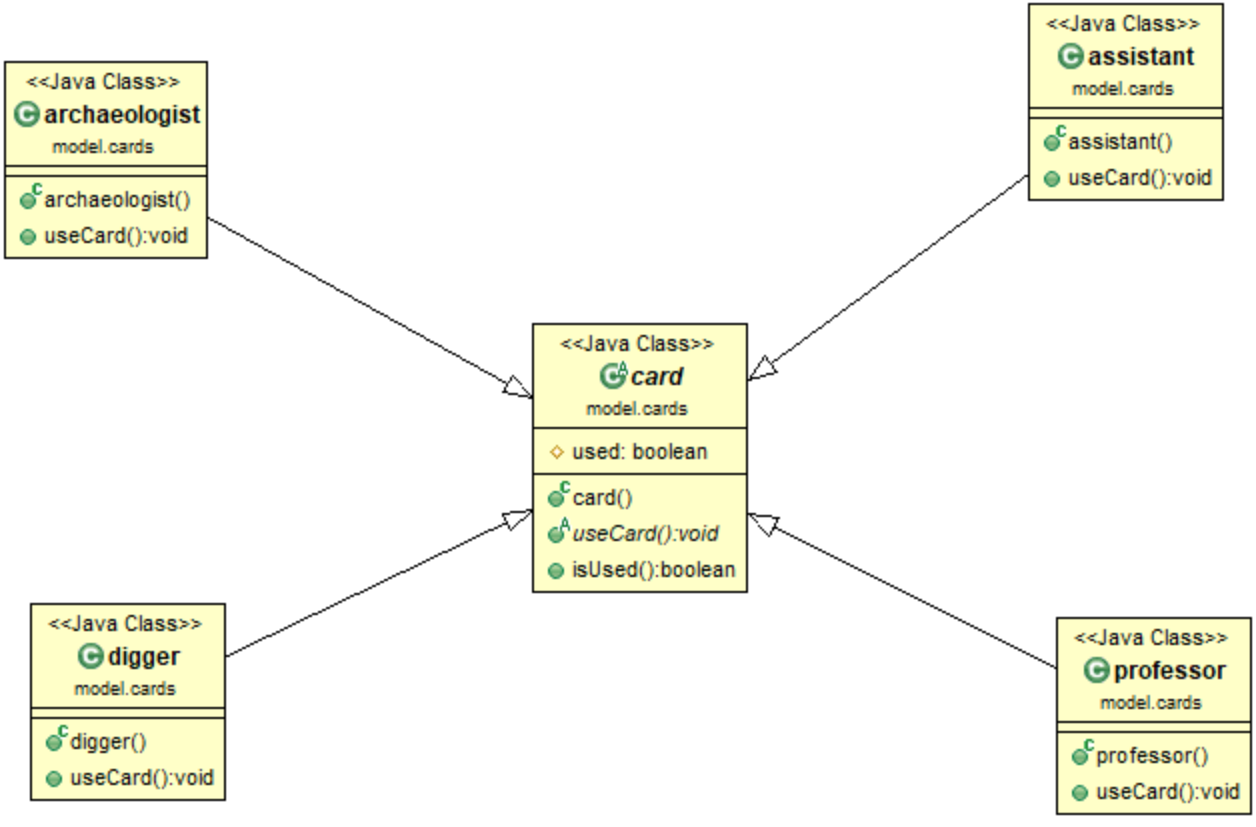
Επιστρεφει τον ΝΙκητη εφοσον το παιχνηδη εχει τελεισει και η περιοχη εισοδου εχει γεμιση. Θα γινει χρησει τις μεθοδου που θα υλοποιηθει στην κλαση Player που υπολογιζει τους ποντους και παιχτη.

- Η Σχεδίαση και οι Κλάσεις του Πακέτου View

Αθτο το πακετο θα αποτελειται απο μια κλαση που θα δημιουργει ενα frame και μεσα σε αυτο ενα panel.Μεσα στο πανελ θα εχουμε το ταμπλε με τις περιοχες που θα ειναι πανελς και αυτες επιτα θα εχουμε ενα πανελ που θα ειναι η περιοχη του παιχτη που παιζει τωρα και θα αλλαζει οταν ο παιχτης πατησει Next Player. Στο χωρο του παιχτη θα εχουμε ενα πανελ με της καρτες του παιχτη επισης θα δειχνουμαι πληροφοροιες για τον Παιχτη Ονομα, Ποντη, Tiles που εχει μαζεψει. Καθε φορα που αλλαζει η σειρα οι πληροφοριες θα αλλαζουν και θα εμφανιζονται για τον επομενο παιχτη. Επισης θα υπαρχει κουμπη για το Quit και το New Game.

- Η Αλληλεπίδραση μεταξύ των κλάσεων – Διαγράμματα UML


<<Java Enumeration>>
skeletonParts
model.tiles
UPBIG: skeletonParts
DOWNBIG: skeletonParts
UPSMALL: skeletonParts
DOWNSMALL: skeletonParts
skeletonParts()
-part
0..1
<<Java Class>>
Skeletons
model.tiles
Skeletons(skeletonParts)
getPart():skeletonParts
<<Java Enumeration>>
marbleTypes
model.tiles
Caryatids: marbleTypes
Sphinxes: marbleTypes
marbleTypes()
<<Java Class>>
Amphoras
model.tiles
Amphoras(tileColor)
getColor():tileColor
-type
0..1
<<Java Interface>>
tile
model.tiles
toString():String
<<Java Class>>
Mosaics
model.tiles
Mosaics(tileColor)
getColor():tileColor
-tiles
0..*
<<Java Class>>
bag
model.tiles
bag()
init_tiles():void
isEmpty():boolean
addTile(tile):void
removeTile():void
size():int
clearAll():void
return4Tiles():ArrayList<tile>
remove4Tiles():void
randomizeBag():void
<<Java Class>>
MarbleStatues
model.tiles
MarbleStatues(marbleTypes)
getType():marbleTypes
-col
0..1
<<Java Enumeration>>
tileColor
model.tiles
GREEN: tileColor
BROWN: tileColor
PURPLE: tileColor
YELLOW: tileColor
BLUE: tileColor
RED: tileColor
tileColor()
-col
0..1
<<Java Class>>
Landslides
model.tiles
Landslides()


<<Java Class>>
archaeologist
model.cards
archaeologist()
useCard():void
<<Java Class>>
assistant
model.cards
assistant()
useCard():void
<<Java Class>>
card
model.cards
used: boolean
card()
useCard():void
isUsed():boolean
<<Java Class>>
digger
model.cards
digger()
useCard():void
<<Java Class>>
professor
model.cards
professor()
useCard():void