Πανεπιστήμιο Κρήτης –Τμήμα Επιστήμης Υπολογιστών

HY252– Αντικειμενοστρεφής Προγραμματισμός

Διδάσκων: Ι. Τζίτζικας

Χειμερινό Εξάμηνο 2020-2021

ΑΝΑΦΟΡΑ ΓΙΑ ΤΟ PROJECT

Μιχάλης Καραγιαννάκης

4355

9/01/2020

Περιεχόμενα

1. Εισαγωγή

Η υλοποίηση του project θα βασιστεί πάνω στο μοντέλο MVC (Model-View-Controller), με στόχο ο Controller να συνδέει το Model με το View. Στη συνέχεια της αναφοράς θα υπάρξει ανάλυση του πακέτου Model , του πακέτου Controller και στο τέλος για το πακέτο View. Τέλος θα γίνει ανάλυση των παραπάνω πακέτων και μέσω διαγραμμάτων UML θα εξηγήσω την αλληλεπίδραση μεταξύ των κλάσεων.

2. Η Σχεδίαση και οι Κλάσεις του Πακέτου Model

Σε αυτό το πακέτο θα περιέχονται η διεπαφή Tile, οι κλάσεις FindingTile και LandslideTile, κλάσεις που κληρονομούν την FindingTile, η κλάση Player, ή κλάση Character και οι κλάσεις που την κληρονομούν, οι κλάσεις AmphoraTileColor, PlayerColor και MosaicTileColor για το χρώμα κάποιων Tile και των χαρακτήρων και τέλος οι κλάσεις Bag και Board.

Classes

Tile Interface

Αρχικά δημιουργώ την διεπαφή Tile, η οποία ορίζει μεθόδους που πρέπει κάθε είδος Tile να υλοποιεί.

Το interface παρέχει τις παρακάτω μεθόδους:

1. public void setCategory(String category); Transformer

   Sets the tile's category.

2. public String getCategory(); Accessor

   Returns the tile's category.

Στη συνέχεια έχουμε τις κλάσεις LandslideTile και FindingTile που υλοποιούν την διεπαφή Tile.

Class LandslideTile

Εδώ θα γίνει αναφορά στα attributes και στις μεθόδους της κλάσης LandslideTile.

Attributes:

1. private String category; //The category of a Tile.

Methods:

1. public LandslideTile(); Constructor

   Creates a new LandslideTile and sets its category.

2. public void setCategory(String category); Transformer

   Sets the tile's category.

3. public String getCategory(); Accessor

   Returns the tile's category.

Class FindingTile

Εδώ θα γίνει αναφορά στα attributes και στις μεθόδους της κλάσης FindingTile.

Attributes:

1. private String category; //The category of a Tile.

Methods:

1. public FindingTile(String category); Constructor

   Creates a new FindingTile and sets its category.

2. public void setCategory(String category); Transformer

   Sets the FindingTile's category.

3. public String getCategory(); Accessor

   Returns the FindingTile's category.

Class AmphoraTileColor

Είναι μια enum class που περιέχει τα χρώματα των AmphoraTile(BLUE, BROWN, RED, GREEN, YELLOW, PURPLE).

Class AmphoraTile

Εδώ θα γίνει αναφορά στα attributes και στις μεθόδους της κλάσης AmphoraTile.

Attributes:

1. private AmphoraTileColor color; //The color of an AmphoraTile.

Methods:

1. public AmphoraTile(AmphoraTileColor color); Constructor

   Constructs a new instance of AmphoraTile, sets the color and via the superclass(FindingTile) sets the category=AmphoraTile.

2. public AmphoraTileColor getColor(); Accessor

   Returns the Amphoratile's color.

3. public void setColor(AmphoraTileColor color); Transformer

   Sets the Amphoratile's color.

Class StatueTile

Εδώ θα γίνει αναφορά στα attributes και στις μεθόδους της κλάσης StatueTile.

Attributes:

1. private String category; //The category of a StatueTile.

Methods:

1. public StatueTile(String category); Constructor

   Passes the category of the StatueTile to the superclass(FindingTile).

2. public String getCategory(); Accessor

   Returns the Statuetile's category.

3. public void setCategory(String category); Transformer

   Sets the Statuetile's category.

Στη συνέχεια έχουμε τις κλάσεις CaryatidTile και SphinxTile που είναι υποκλάσεις της StatueTile.

Classes CaryatidTile -SphinxTile

Αυτές οι κλάσεις κάνουν extend την StatueTile και μέσω της εντολής super αποκτούν πρόσβαση στην κλάση StatueTile και αρχικοποιούν το αντίστοιχο category κάθε φορά.

Class MosaicTileColor

Είναι μια enum class που περιέχει τα χρώματα των MosaicTile(GREEN,RED,YELLOW).

Class MosaicTile

Εδώ θα γίνει αναφορά στα attributes και στις μεθόδους της κλάσης MosaicTile.

Attributes:

1. private MosaicTileColor color; //The color of a MosaicTile.

Methods:

1. public MosaicTile(MosaicTileColor color); Constructor

   Constructs a new instance of MosaicTile, sets the color and via the superclass(FindingTile) sets the category=MosaicTile.

2. public MosaicTileColor getColor(); Accessor

   Returns the MosaicTile's color.

3. public void setColor(MosaicTileColor color); Transformer

   Sets the MosaicTile's color.

Class SkeletonTile

Εδώ θα γίνει αναφορά στα attributes και στις μεθόδους της κλάσης SkeletonTile. Η κλάση SkeletonTile περιέχει πέρα απο το String category, και άλλα δύο String , το size και το body_part τα οποία διαχωρίζουν τα πλακίδια των σκελετών κάτι που θα είναι χρήσιμο και στον υπολογισμό των πόντων των παικτών.

Attributes:

1. private String category; //The category of a SkeletonTile.
2. private String size; //The size of the skeleton.
3. private String body_part; //The body part of the skeleton.

Methods:

1. public SkeletonTile(String size , String body_part); Constructor

   Constructs a new instance of SkeletonTile, sets the size and the body_part and via the superclass(FindingTile) sets the category=SkeletonTile.

2. public String getSize(); Accessor

   Returns the size of the skeleton.

3. public String getBodyPart(); Accessor

   Returns the body part of the skeleton.

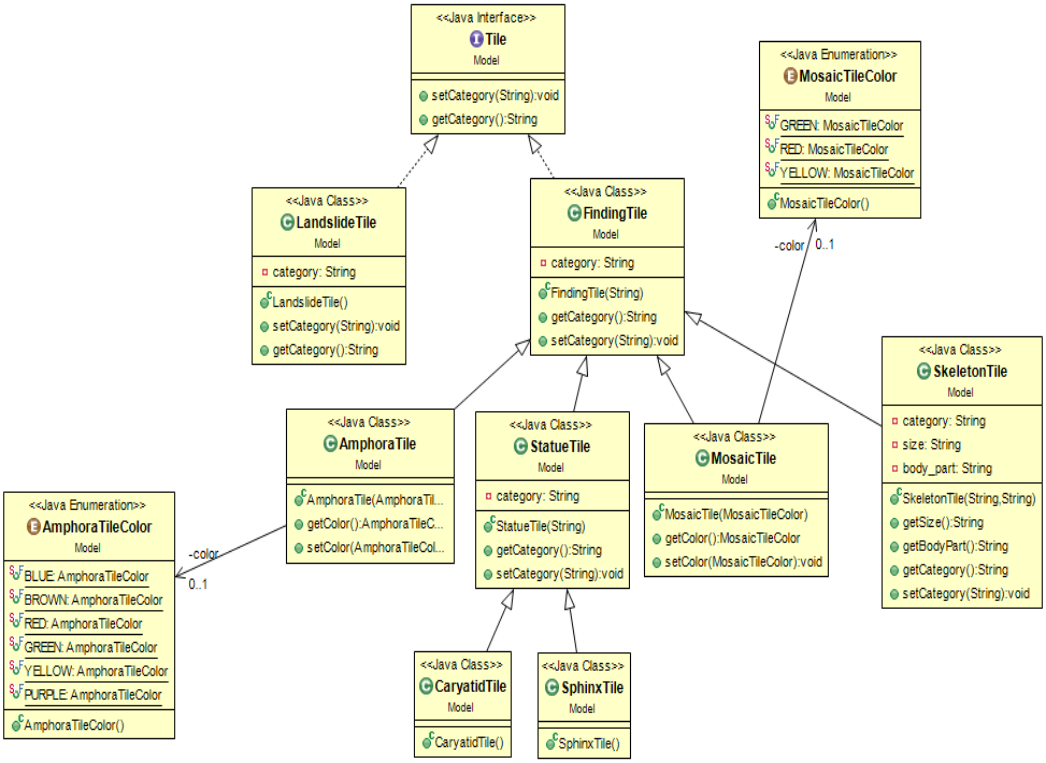
4. public String getCategory(); Accessor

   Returns the tile's category.

5. public void setCategory(String category); Transformer

   Sets the tile's category.

Παρακάτω γίνεται μια αναπαράσταση των κλάσεων που αναλύθηκαν παραπάνω και τις σχέσεις μεταξύ τους, μέσω ενός UML διαγράμματος.



Class Bag

Αυτή η κλάση αντιπροσωπεύει την σακούλα με όλα τα πλακίδια στην αρχή του παιχνιδιού. Μέσω αυτής δημιουργείται η σακούλα(ArrayList) με τα πλακίδια (Tiles), αρχικοποιούνται, ελέγχει αν άδειασε η σακούλα, αφαιρεί τα πλακίδια που κάθε φορά επιλέχθηκαν τυχαία από τους παίκτες και μια μέθοδο που γυρνάει την σακούλα(ArrayList).

Attributes:

1. private ArrayList<Tile> tiles; //A Tile ArrayList.

Methods:

1. public Bag(); Constructor

Creates an ArrayList that contains all the tiles and with that way the Bag is created.

2. public void init_tiles(); Transformer

   Initializes the tiles.

3. public boolean isEmpty(); Observer

   Returns true if this list contains no elements.

4. public void removeTile(Tile tile); Transformer

   Removes a tile from the list.

5. public ArrayList<Tile> getBag(); Accessor

   Returns the whole bag(the ArrayList that consists of the Tiles).

Class PlayerColor

Είναι μια enum class που περιέχει τα χρώματα που διαχωρίζουν τους παίκτες(YELLOW,RED,BLUE,BLACK).

Class Player

Αυτή η κλάση είναι υπεύθυνη για ότι έχει να κάνει με τον παίκτη. Μέσω αυτής δημιουργείται ο παίκτης , μπορούμε να δώσουμε όνομα στον παίκτη, να πάρουμε το όνομα του, να ελεγξουμε αν έπαιξε ο παίκτης τουλάχιστον μία φορά και μέθοδο που επιστρέφει τα Tile του παίκτη. Επίσης μπορεί να προσθέσει και να αφαιρέσει Tile απο την συλλογή του και υπολογίζει τους πόντους του.

Attributes:

1. private String name; // The name of the player.
2. private boolean hasPlayed; //If the player has played atleast once or not.
3. private ArrayList<Tile> player_tiles; //A Tile ArrayList for the player.
4. private int points; // The points of the player.
5. private PlayerColor color; //The color that a player has.
6. private ArrayList<Character> players_characters; // A Character Arraylist for the characters of the player.

7. private boolean finished; //If the player has finished their turn or not.
8. private ArrayList<JLabel> playerTileLabels; //A JLabel ArrayList that contains the the JLabels of the tiles that the player has.

Methods

1. public Player(PlayerColor color , String newName); Constructor

   Creates a player by setting its name,color and hasPlayed=false. Also, it creates ArrayLists for the tiles of the player and their characters.

2. public String getName(); Accessor

   Returns the player's name.

3. public void setName(String newName); Transformer

   Sets the player's name.

4. public boolean has_Played(); Observer

   Returns if a player has played at least once.

5. public ArrayList<Tile> getTiles(); Accessor

   Returns the tiles that are in the collection of the player.

6. public addTile(Tile tile); Transformer

   Adds a tile to the list.

7. public void removeTile(Tile tile); Transformer

   Removes a tile from the list.

8. public int pointCalculator(ArrayList<Tile> tiles); Transformer

   Calculates the points of the player.

9. public PlayerColor getColor(); Accessor

   Returns the player's color.

10. public ArrayList<Character> getCharacter(); Accessor

    Returns the characters of the player.

11. public boolean has_finished(); Transformer

    Returns if the player has finished their turn.

12. public addCharacter(Character character); Transformer

    Adds a character to the list.

13. public removeCharacter(Character character); Transformer

    Removes a character from the list.

14. public ArrayList<JLabel> getLabels(); Accessor

    Returns the tile labels of the player.

15. public int getCaryatid(); Transformer

    Counts the player's total CaryatidTiles.

16. public givePoints(int points); Transformer

    Adds points to the player.

17. public int getPoints(); Accessor

    Returns the points of the player.

18. public getSphinx(); Transformer

    Counts the player's total SphinxTiles.

Class Board

Αυτή η κλάση είναι υπεύθυνη για το ταμπλό του παιχνιδιού. Μέσω αυτής δημιουργούνται οι 5 περιοχές για τα Tile, αρχικοποιείται το παιχνίδι, μέθοδοι για το όνομα του παίκτη που έχει σειρά, μέθοδοι για τον τελευταίο παίκτη που έπαιξε, έλεγχος αν ο παίκτης τέλειωσε και έλεγχος για το αν τέλειωσε το παιχνίδι.

Attributes:

1. private ArrayList<MosaicTile> MosaicsSortingArea; //A MosaicTile ArrayList.
2. private ArrayList<StatueTile> MarbleStatuesSortingArea; //A StatueTile ArrayList.
3. private ArrayList<AmphoraTile> AmphorasSortingArea; //An AmphoraTile ArrayList.
4. private ArrayList<SkeletonTile> SkeletonsSortingArea; //A SkeletonTile ArrayList.
5. private ArrayList<LandslideTile> EntranceSpaces; //A LandslideTile ArrayList.
6. private String currentPlayer; //The name of the player that is currently playing.
7. private String last_player; //The name of the last player that played.

Methods:

1. public Board(); Constructor

   Creates the ArrayLists of the five tile areas.

2. public void setTurn(ArrayList <Player> players); Transformer

   Sets the player's turn.(which player has the turn to play).

3. public String getTurn(); Accessor

   Return the players' name whose turn is to play.

4. public ArrayList<MosaicTile> getMosaicArea; Accessor

   Returns the tiles of the MosaicsSortingArea.

5. public ArrayList<StatueTile> getStatueArea(); Accessor

   Returns the tiles of the MarbleStatuesSortingArea.

6. public ArrayList<AmphoraTile> getAmphoraArea(); Accessor

Returns the Tiles of the AmphoraSortingArea.

7. public ArrayList<SkeletonTile> getSkeletonArea(); Accessor

   Returns the Tiles of the SkeletonsSortingArea.

8. public ArrayList<LandslideTile> getLandslideArea(); Accessor

   Returns the Tiles of the EntranceSpaces.

9. public String getLastPlayer(); Accessor

   Returns the last player that played.

10. public void setLastPlayer(String name); Transformer

    Sets the last player.

11. public boolean checkIfPlayerFinished(Player p); Observer

    Checks if a player has finished their turn.

12. public boolean End(); Observer

    Checks if the game ends.

Abstract Class Character

Η αφηρημένη κλάση Character αντιπροσωπεύει τις κάρτες χαρακτήρων που μοιράζονται στους παίκτες. Υπάρχουν 4 διαφορετικοί χαρακτήρες άρα και 4 υποκλάσεις που κάθε μια αποτελεί ένα χαρακτήρα. Έχει μια μεταβλητή η οποία μας πληροφορεί για το αν έχει χρησιμοποιηθεί η κάρτα χαρακτήρα ήδη(boolean isUsed) και μια μεταβλητή η οποία αντιπροσωπεύει τον παίκτη που έχει τον χαρακτήρα.

Παρακάτω γίνεται αναφορά στα attributes και στις μεθόδους της κλάσης Character.

Attributes:

1. protected boolean isUsed; //Its value shows if the character has been used or not.
2. protected Player player; //An instance of a player.

Methods:

1. public Character(boolean isUsed , Player player); Constructor

   It will create a character through its subclasses and sets isUsed and player.

2. public boolean getIsUsed(); Accessor

   Returns if the character has been used already.

3. public Player getPlayer(); Accessor

   Returns the player that owns the character card.

Στη συνέχεια ακολουθούν τα είδη χαρακτήρων δηλαδή τα subclass της κλάσης Character.(Archaeologist,Assistant,Digger,Professor).

Class Archaeologist

Εδώ θα γίνει αναφορά στις μεθόδους της κλάσης Archaeologist.

Methods:

1. public Archaeologist(boolean isUsed , Player player); Constructor

   Creates an instance of Archaeologist and through the superclass it sets two characteristics: isUsed and player.

2. public boolean getIsUsed(); Accessor

   Returns if the character has been used already.

3. public Player getPlayer(); Accessor

   Returns the player that owns the character card.

Class Assistant

Εδώ θα γίνει αναφορά στις μεθόδους της κλάσης Assistant.

Methods:

1. public Assistant(boolean isUsed , Player player); Constructor

Creates an instance of Assistant and through the superclass it sets two characteristics: isUsed and player.

2. public boolean getIsUsed(); Accessor

   Returns if the character has been used already.

3. public Player getPlayer(); Accessor

   Returns the player that owns the character card.

Class Digger

Εδώ θα γίνει αναφορά στις μεθόδους της κλάσης Digger.

Methods:

1. public Digger(boolean isUsed , Player player); Constructor

   Creates an instance of Digger and through the superclass it sets two characteristics: isUsed and player.

2. public boolean getIsUsed(); Accessor

   Returns if the character has been used already.

3. public Player getPlayer(); Accessor

   Returns the player that owns the character card.

Class Professor

Εδώ θα γίνει αναφορά στις μεθόδους της κλάσης Professor.

Methods:

1. public Professor(boolean isUsed , Player player); Constructor

   Creates an instance of Professor and through the superclass it sets two characteristics: isUsed and player.

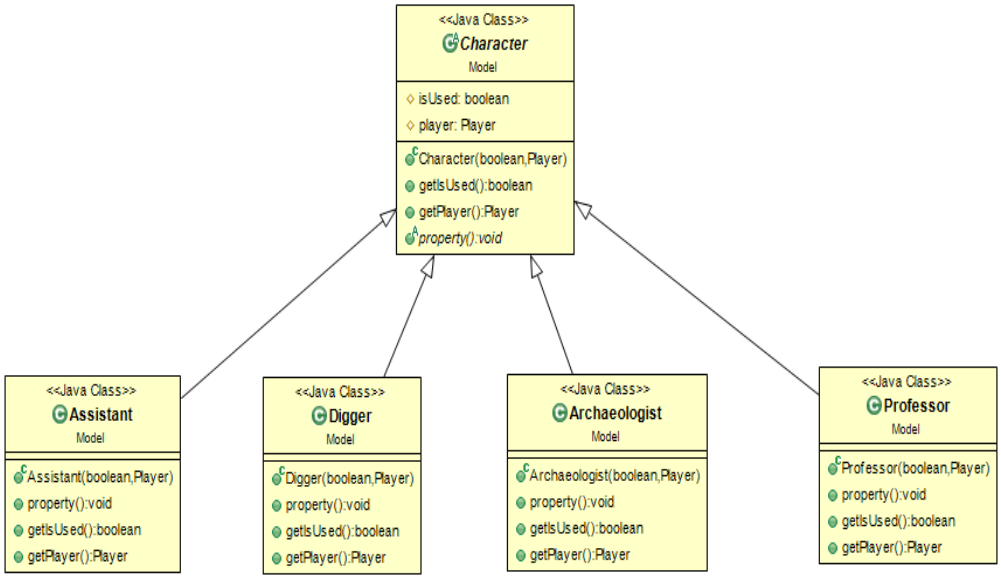
2. public boolean getIsUsed(); Accessor

   Returns if the character has been used already.

3. public Player getPlayer(); Accessor

   Returns the player that owns the character card.

Εδώ μια αναπαράσταση σε UML της κλάσης Character και των υποκλάσεων της.



Β Φάση

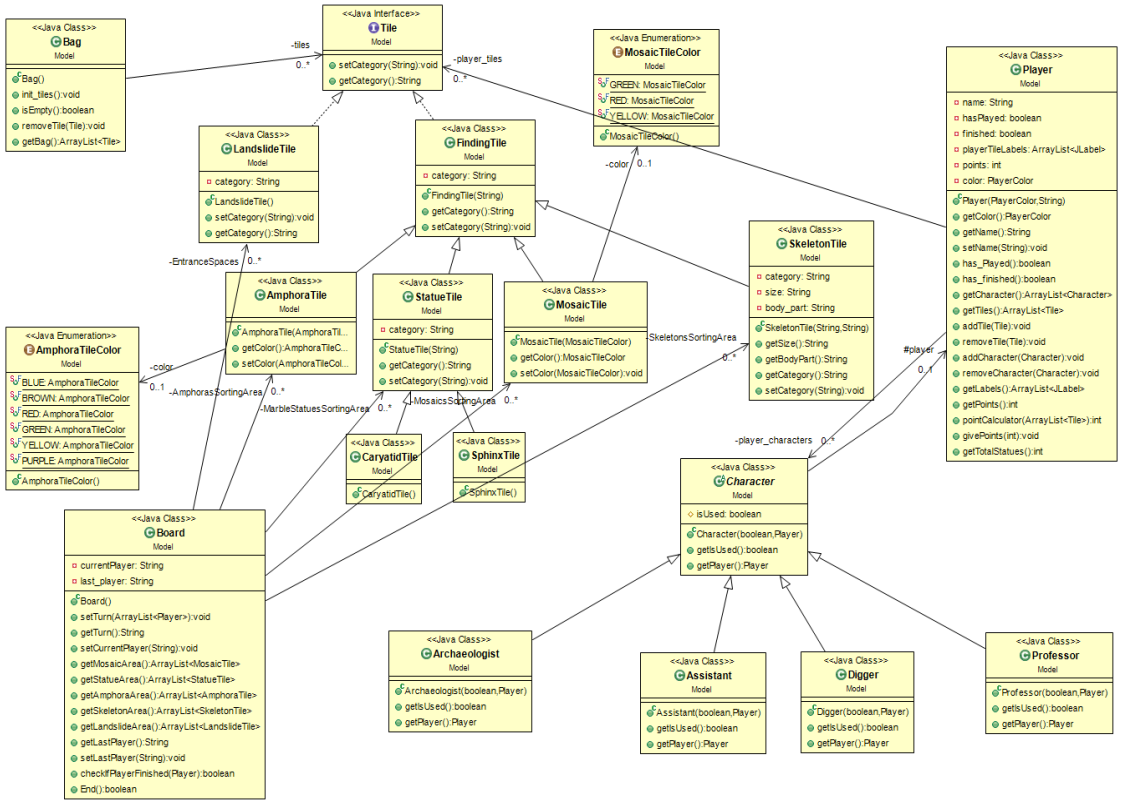
Από το πακέτο Model χρησιμοποιήθηκαν αρκετά πράγματα από την Φάση Α. Αρχικά, στις κλάσεις που έχουν να κάνουν με τα είδη των Tile χρησιμοποιήθηκαν κυρίως οι Constructors τους για την δημιουργία του πλακιδίου, εκτός απο ειδικές κλάσεις που ήθελαν και κάτι παραπάνω. Συγκεκριμένα στην AmphoraTile και στην MosaicTile ήταν πάρα πολύ χρήσιμη η μέθοδος getColor() σε πολλά πράγματα στην υλοποίηση του παιχνιδιού όπως στην αρχικοποίηση των πλακιδίων και του Bag, στην δημιουργία των σωστών JButton και JLabel για την απεικόνιση τους στο GUI και στην βαθμολόγηση των παικτών. Στην SkeletonTile χρησιμοποιήθηκαν οι getSize() και getBodyPart() οι οποίες βοήθησαν πολύ στο να ξέρω το είδος σκελετού που δημιουργώ αλλά και βρίσκω στην σακούλα ή στα πλακίδια του παίκτη.

Από την κλάση Bag χρησιμοποίησα την Constructor και την init_tiles() για την αρχικοποίηση της σακούλας και των πλακιδίων, κάτι πολύ σημαντικό στο παιχνίδι αλλά και την getBag() για να έχω πρόσβαση στην Bag. Οι υπόλοιπες δεν ήταν τόσο χρήσιμες.

Από την κλάση Player ,που αποτελεί μια από τις πιο βασικές(αν όχι η πιο βασική) σε όλο το πακέτο του Model , χρησιμοποίησα πάρα πολλές. Από μεθόδους της Φάσης Α χρησιμοποίησα πολύ την Constructor , την getName(), την getTiles(),την pointCalculator() και την getCharacter(). Οι addTile και removeTile δεν φάνηκαν τόσο χρήσιμες αφού απλά μέσω του ArrayList έκανα τα add,get,remove κτλπ. Στην Φάση Β πρόσθεσα πολλές μεθόδους οι οποίες βοήθησαν την υλοποίηση μου όπως οι addCharacter, removeCharacter, getLabels, getPoints, givePoints, getCaryatid και getSphinx. Οι πρώτες δύο για το ArrayList με τους χαρακτήρες, η τρίτη για τα Labels του κάθε παίκτη και οι άλλες τέσσερις για τους βαθμούς.

Από την κλάση Board χρησιμοποίησα σχεδόν όλες τις μεθόδους με πιο σημαντικές την set και getTurn που καθόριζουν την σειρά των παικτών, τους getters των περιοχών του ταμπλό, και την End που ελέγχει πότε τελειώνει το παιχνίδι.

Τέλος όσον αφορά τους χαρακτήρες δεν χρησιμοποίησα την isUsed και ούτε την μέθοδο property(), επειδή την ιδιότητα τελικά την υλοποίησα στο πακέτο View. Για την αρχικοποίηση των χαρακτήρων για κάθε παίκτη, χρησιμοποιήθηκε η Constructor κάθε χαρακτήρα.



3. Η Σχεδίαση και οι Κλάσεις του Πακέτου Controller

Το πακέτο Controller αποτελεί τον εγκέφαλο επικοινωνίας και συντονισμού του παιχνιδιού. Είναι υπεύθυνο για την δημιουργία ενός νέου παιχνιδιού αλλά και για την δημιουργία στιγμιοτύπων παικτών . Επίσης με την δημιουργία object των κλάσεων Bag και Board, αρχικοποιεί και το ταμπλό και την σακούλα με τα πλακίδια ενώ έχει μεθόδους για την σειρά που παίζουν οι παίκτες και για την παραλαβή πλακιδίων από τους παίκτες. Φυσικά είναι υπεύθυνο και για την σύνδεση του Model και του View.

Παρακάτω θα γίνει αναφορά στα attributes και στις μεθόδους της κλάσης.

Attributes:

1. private ArrayList<Player> players; // A Player ArrayList that consists of all the players.
2. private Player P1,P2,P3,P4; //The players of the game.
3. private boolean notstarted,empty_board //Variables that check if the game has started and if the board is empty respectively.
4. private Bag alltiles; // The bag that contains all the tiles at the beginning of the game.
5. private Board board; // The board that the game will be played on.
6. private ArrayList<Player> standings; //A Player ArrayList that has the players placed in order in relation to their points.

Methods:

1. public Controller(); Constructor

   Constructs a new Controller and sets up the game for the players to play.

2. public void init_character_cards(); Transformer

   Initializes the character cards in the beginning of the game.

3. public String seeTurn(); Accessor

   Returns which player has the turn to play.

4. public void EndTurn(); Transformer

Ends the turn of a player if they player pushes the “End Turn” button.

5. public void DrawTiles(); Transformer

   Draws tiles for the player if the player pushes the “Draw Tiles” button.

6. public Player firstToPlay(); Accessor

   Returns the player that was selected randomly, to play first.

7. public Player Winner(); Accessor

   Finds the winner of the game.

8. public boolean hasStarted(); Accessor

   Returns if the game has started or not.

9. public void setHasStarted(boolean notstarted); Transformer

   Sets the state of the game.

10. public ArrayList<Player> getPlayers(); Accessor

    Returns the players that participate in the game.

11. public ArrayList<Tile> getPlayersTiles(Player player); Accessor

    Returns the tiles of a specific player.

12. public Bag getAllTiles(); Accessor

    Returns the tiles that are in the bag.

13. public void getSetTurn(); Accessor

    Sets the turn via setTurn() that is implemented in board Class.

14. public String getBoardTurn(); Accessor

    Returns the method getTurn() that is implemented in board Class.

15. public void Points(); Transformer

    Concludes the calculation process with the addition of the StatueTile points.

16. public void Standings(); Transformer

Sets the order of the players in relation to their points.

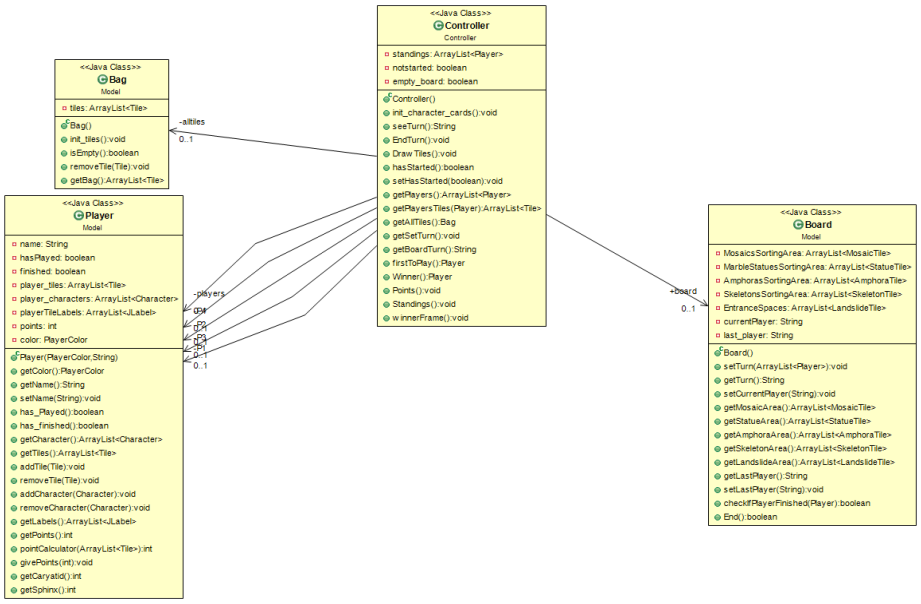
17. public void winnerFrame(); Transformer

Creates frame,panels and labels that have to do with the standings.

Β Φάση

Στο πακέτο Controller έχουν προστεθεί πολλές νέες μέθοδοι όπως και στην κλάση Player του πακέτου Model. Μέθοδοι που χρησιμοποίησα απο την Φάση Α είναι η Controller() για την αρχικοποίηση του παιχνιδιού και οι init_character_cards, seeTurn, EndTurn, DrawTiles, firstToPlay και Winner. Οι τρείς απο αυτές έχουν να κάνουν με την σειρά του παίκτη , η μια με την αρχικοποίηση των καρτών των χαρακτήρων, μια με το τράβηγμα πλακιδίων και η τελευταία με τον νικητή του παιχνιδιου. Στην Φάση Β προστέθηκαν σημαντικές μέθοδοι όπως τις getPlayers, getPlayersTiles, getAllTiles, getSetTurn, getBoardTurn που όλες επιστρέφουν κάτι από κλάσεις της Model. Οι 2 πρώτες επιστρέφουν τους παίκτες και τα Tile του κάθε παίκτη, η getAllTiles επιστρέφει το ArrayList που περιέχει όλα τα tile του παιχνιδιού(Bag) και οι επόμενες δύο έχουν να κάνουν με την σειρά. Τέλος υλοποιήθηκε η Points() η οποία καλεί την pointsCalculator της κλάσης Player , δηλαδη υπολογίζει τους πόντους των παικτών για τα Μωσαικά , τους Αμφορείς και τους Σκελετούς , και στο τέλος υπολογίζει τους βαθμούς για τα Αγάλματα και τους προσθέτει.

Τέλος υλοποιούνται δύο μέθοδοι που έχουν να κάνουν με την εμφάνιση των τελικών αποτελεσμάτων. Η Standings() έχει να κάνει με τον υπολογισμό των αποτελεσμάτων(δηλαδή ποιος νίκησε ,ποιος είναι 2ος κτλπ.) και η winnerFrame() υλοποιεί τα γραφικά που παρουσιάζουν τα αποτελέσματα.



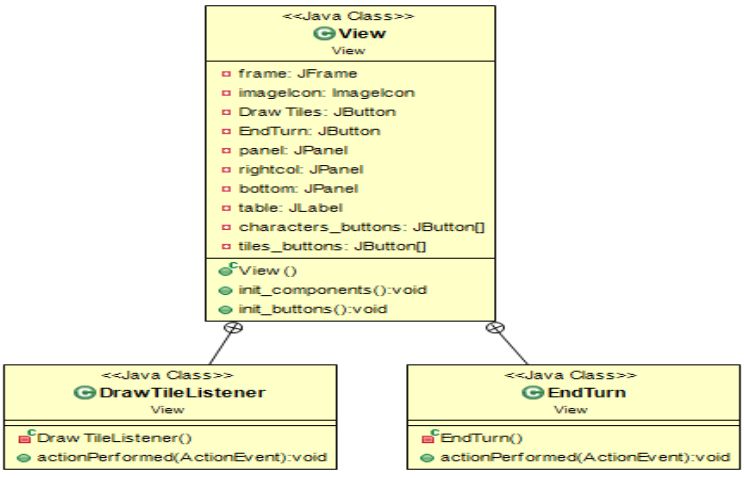
4. Η Σχεδίαση και οι Κλάσεις του Πακέτου View

Το πακέτο View αποτελεί την γραφική απεικόνιση του παιχνιδιού. Θα δημιουργεί αρχικά 1 frame και 3 panel. Το πρώτο panel θα έχει να κάνει με το ταμπλό, το δεύτερο με τα tile που έχει ο παίκτης και το τρίτο panel θα έχει ποιός παίκτης παίζει , τις κάρτες χαρακτήρων του και τα κουμπιά για να τραβήξει tiles και για να τελειώσει την σειρά του. Στο πρώτο panel θα υπάρχει ενα label το οποίο θα βάζει το background image(ταμπλό). Επίσης στο τρίτο panel πέρα από τα κουμπιά DrawTiles και EndTurn θα υπάρχουν label για το ποιός παίκτης παίζει και κουμπιά στους χαρακτήρες έτσι ώστε να μπορεί ο χρήστης να τις χρησιμοποιήσει. Ακόμα θα υπάρχουν 135 κουμπιά για τα πλακίδια έτσι ώστε να μπορούν να οι παίκτες και να τα βάζουν στα κατάλληλα πεδία στο ταμπλό αλλά και να επιλέγουν από αυτά. Όταν ένας παίκτης πατάει την κάρτα χαρακτήρα για να την χρησιμοποιήσει, η κάρτα θα γίνεται γκρι και δεν θα είναι διαθέσιμη για το υπόλοιπο παιχνίδι.

Από μεθόδους υπάρχει η View() η οποία θα δημιουργεί τα γραφικά του ταμπλό και θα αρχικοποιεί κάποια buttons,panels κτλπ. Η init_components() θα αρχικοποιεί κάποια buttons και labels και η init_buttons() θα κάνει set και θα δουλέυει με buttons και labels για κάθε σειρά παίκτη.

Τέλος υπάρχει η κλάση DrawTileListener η οποία είναι υπεύθυνη για την λειτουργία του κουμπιού Draw Tile αν το πατήσει ο παίκτης και η κλάση EndTurn η οποία είναι υπεύθυνη για την λειτουργία του κουμπιού End Turn.

Εδώ φαίνεται το πακέτο View της Phase A σε διάγραμμα UML.

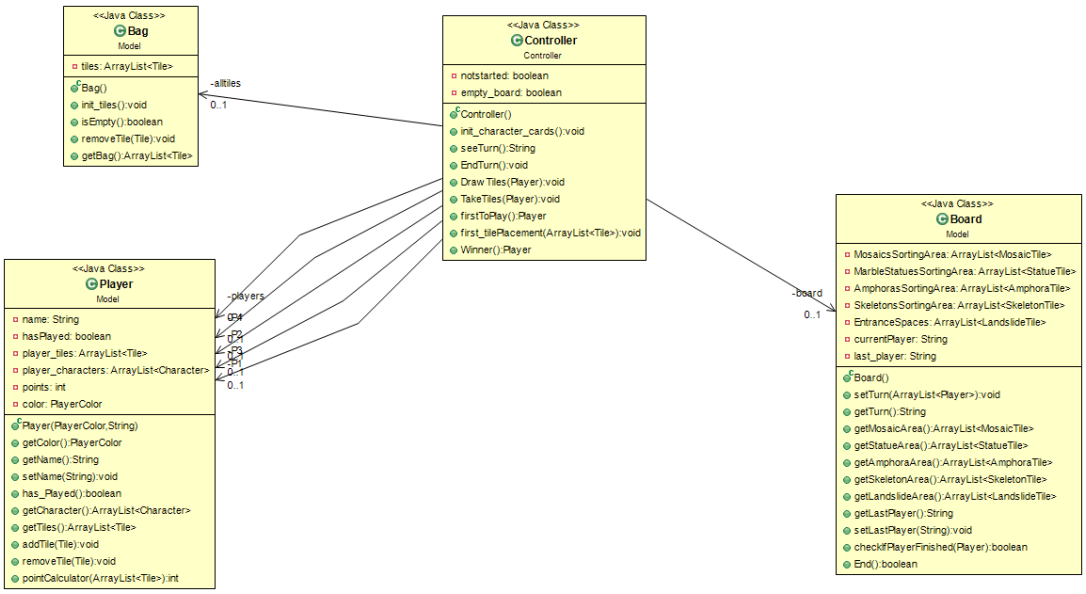


Β Φάση

Στην Β φάση του project έγιναν αλλάγες στο πακέτο View. Στην φάση Α είχα σκεφτεί και είχα αρχίσει να κάνω τα πάντα μόνος μου (manually). Στην φάση Β και με την βοήθεια του WindowBuilder του Eclipse άλλαξε ο τρόπος προσέγγισης μου. Δεν υπάρχουν οι μέθοδοι init_components και init_buttons που είχα στην φάση Α, και έχω φτιάξει τις createGame() και init_characterButtons(). Η createGame περιέχει σχεδόν όλο τον κώδικα του GraphicUI που περιέχει την δημιουργία όλων των panels,labels και buttons για τα γραφικά του παιχνιδιού αλλά και το τι θα κάνει το πρόγραμμα όταν ο παίκτης επιλέξει να πάρει ένα Tile από το Board ή να πατήσει τα κουμπιά DrawTiles και End Turn. Η init_characterButtons έχει να κάνει με τα κουμπιά των χαρακτήρων και τις ιδιότητες τους. Επίσης έχουν αλλάξει και έχουν προστεθεί νέες μεταβλητές στην GraphicUI , JPanels, JButtons , JLabels, μεταβλητές για το position των tile στο board και πολλά flags που βοηθούν στην υλοποίηση κάποιων λειτουργιών.

Όπως και το πλάνο μου στην φάση Α , το GUI αποτελείται από 3 JPanel , το ένα είναι το ταμπλό, το άλλο περιέχει δυο JLabel για τον παίκτη που έχει σειρα και την πρόταση “Use Character” ,τους χαρακτήρες και τα κουμπιά και το τελευταίο για τα πλακίδια του παίκτη.

5. Η Αλληλεπίδραση μεταξύ των κλάσεων – Διαγράμματα UML



Όπως φαίνεται και στο UML διάγραμμα , μέχρι στιγμής στην Φάση Α , συνδέονται τρεις κλάσσεις του Package Model με το package Controller. Η Bag , η Player και η Board.

Η σύνδεση μεταξύ της Bag και της Controller είναι η δημιουργία ενός object Bag μέσα στην Controller το οποίο αποτελεί την σακούλα που περιέχει όλα τα πλακίδια στην αρχή του παιχνιδιού. Μέσω αυτού του object μπορούμε να κάνουμε access την κλάση της Bag και τις μεθόδους της όπως για την αρχικοποίηση των πλακιδίων , την αφαίρεση πλακιδίου από την σακούλα , έλεγχος αν είναι άδεια κτλπ.

Η σύνδεση της κλάσης Board με την Controller γίνεται με την δημιουργία ενός object Board στην Controller, η οποία ουσιαστικά δημιουργεί το ταμπλό. Έτσι έχουμε πρόσβαση στην κλάση Board και σε πολύ σημαντικές μεθόδους που έχουν να κάνουν με τις 5 περιοχές των πλακιδίων, τον παίκτη που έπαιξε τελευταίος ,έλεγχος αν ο παίκτης τέλειωσε και για το αν τέλειωσε το παιχνίδι.

Τέλος, η σύνδεση της Player με την Controller γίνεται με τις μεταβλητές των Player(P1,P2,P3,P4) και με το ArrayList<Player> players. Η μεταβλητές Player αντιπροσωπεύουν κάθε μια , ένα παίκτη , και το παραπάνω ArrayList έχει όλους τους παίκτες μέσα. Με την σύνδεση αυτή , στην Controller αρχικοποιούμε τους παίκτες, τους βάζουμε στο ArrayList και επίσης μπορούμε να χρησιμοποιήσουμε χρήσιμες μεθόδους για κάθε παίκτη από την κλάση

Player όπως να πάρουμε το όνομα του, το χρώμα του,τους χαρακτήρες που έχει, τα Tile του, να προσθέσουμε ή να αφαιρέσουμε Tile από την συλλογή του και τέλος να υπολογίσουμε τους πόντους του στο τέλος του παιχνιδιού.

Το πακέτο View μέχρι στιγμής δεν συνδέεται με το Model ή με το Controller, απλά έχει imported το πακέτο Controller.

Β Φάση

Πέρα από τα παραπάνω που ειπώθηκαν στην Α φάση, τώρα πλέον έχει υλοποιηθεί και η View. Η View επικοινωνεί με κλάσεις και μεθόδους του πακέτου Model μέσω ενός object του Controller. Έτσι αρχικοποιεί το παιχνίδι μέσω της Constructor του Controller και έχει πρόσβαση σε χρήσιμες μεθόδους που έχει η Controller ,που μερικές καλούν μεθόδους της Model για την πιο εύκολη επικοινωνία των πακέτων, κάτι που βοηθάει πολύ στην υλοποίηση του παιχνιδιού.

6. Λειτουργικότητα(Β Φάση)

Έχω καταφέρει να υλοποιήσω όλες τις λειτουργίες που έπρεπε στην φάση Β. Αυτές οι λειτουργίες είναι οι εξής:

- **Αρχικοποίηση παικτών** ,που το κάνω μέσω του Constructor της Controller.
- **Αρχικοποίηση ταμπλό** ,που γίνεται με την δημιουργία object της κλάσης Board στην κλάση Controller.
- **Αρχικοποίηση στοίβας καρτών**, που γίνεται με την δημιουργία object της κλάσης Bag στην κλάση Controller.
- **Τήρηση Σειράς** δηλαδή να κάνει πρώτα Draw Tiles ,μετά να επιλέγει και μετά να ενεργοποιεί την ιδιότητα ενός χαρακτήρα αν θέλει, γίνεται στην κλάση GraphicUI του πακέτου View.
- **Τράβηγμα πλακιδίων απο σακούλα και τοποθέτηση στις αντίστοιχες περιοχές** ,που γίνεται μέσω της μεθόδου DrawTiles στην Controller και στην κλάση GraphicUI..
- **Τράβηγμα πλακιδίων από το ταμπλό** , που υλοποιείται στην GraphicUI του View μέσω μεθόδων actionPerformed.

- **Χρήση καρτών χαρακτήρα** ,που υλοποιείται στην κλάση GraphicUI, στην μέθοδο init_character_buttons.

- **Υπολογισμός πόντων** : Ο υπολογισμός των πόντων για τα Μωσαικά ,τους Αμφορείς και τους Σκελετούς γίνεται στην μέθοδο pointsCalculator της κλάσης Player ενώ για τα αγάλματα και το τέλος του υπολογισμού σε μεθόδους στην Controller.

- **Έλεγχος για περιοχή κατολίσθησης-Τέλος παιχνιδιού** : Ο έλεγχος αυτός υλοποιείται στην κλάση GraphicUI , τερματίζει το παιχνίδι και βγάζει τον νικητή σε ένα νέο παράθυρο.

7. Συμπεράσματα

Η εργασία ήταν πολύ ενδιαφέρον,απαιτητική και εκπαιδευτική. Με βοήθησε πολύ στην κατανόηση δημιουργίας ενός GUI και είναι από τα πιο βοηθητικά και πρακτικά πράγματα που έχουμε κάνει μέχρι τώρα στη σχολή. Λόγω αυτού του project, σίγουρα θα επιχειρήσω κάτι παρόμοιο και μόνος στον ελεύθερο μου χρόνο.

Ένα πρόβλημα που είχα συναντήσει στην εργασία ήταν πως τα πλακίδια αντί να εμφανίζονται στο ταμπλό αμέσως, έπρεπε να πηγαίνω τον κέρσορα του ποντικιού από πάνω για να εμφανιστούν. Αλλά λύθηκε με την αλλαγή σειράς κάποιων εντολών που είχαν να κάνουν με το panel και τα button.