Πανεπιστήμιο Κρήτης –Τμήμα Επιστήμης Υπολογιστών

HY252– Αντικειμενοστρεφής Προγραμματισμός

Διδάσκων: Ι. Τζίτζικας

Χειμερινό Εξάμηνο 2020-2021

AMPHIPOLIS

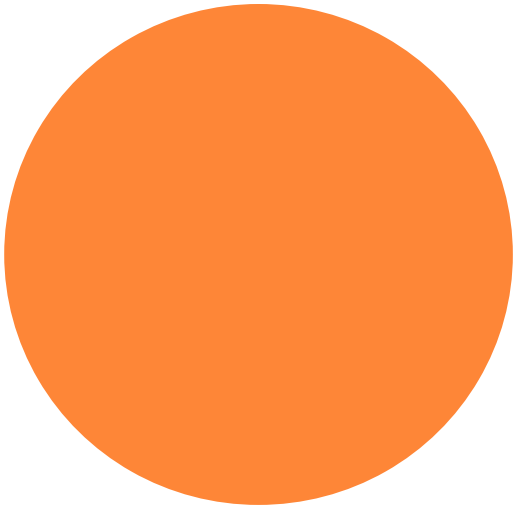
Υλοποίηση του επιτραπέζιου παιχνιδιού “Αμφίπολη” σε java

Όνομα/Επίθετο: Δημήτρης Βλάχος

Αριθμός Μητρώου :Csd_4492

Ημερομηνία :13/1/2021

Περιεχόμενα



Εισαγωγή

Για την ανάπτυξη του Amphipolis χρησιμοποιήθηκε το μοντέλο M.V.C (Model View Controller).Κατά αυτό το μοντέλο η υλοποίηση της εφαρμογής διαχωρίζεται στα παρακάτω πακέτα:

- Την Γραφική διεπαφή (View)
- Τον πυρήνα του παιχνιδιού,που περιέχει τους μηχανισμούς του (Model)
- Τον μηχανισμό διαχείρισης των ενεργειών του παιχνιδιού με την γραφική απεικόνιση (Controller)

Στις επόμενες ενότητες θα γίνει λεπτομερής αναφορά στις υλοποιήσεις των προαναφερθέντων πακέτων ,στις κλάσεις που περιέχουν με τα εκάστοτε χαρακτηριστικά τους και τις μεθόδους τους.

Η Σχεδίαση και οι Κλάσεις του Πακέτου Model

- Το πακέτο Model διαχωρίζεται σε επί μέρους πακέτα τα οποία στοχεύουν να υλοποιήσουν κάθε πτυχή των εργασιών του παιχνιδιού. Τα πακέτα είναι :
- Board: Περιέχει την κλάση Board που είναι υπεύθυνη για τις λειτουργίες του ταμπλό.
- Cards: Περιέχει κλάσεις που υλοποιούν τις λειτουργίες των καρτών χαρακτήρων.
- Player: Περιέχει την κλάση Player η οποία χειρίζεται τις λειτουργίες που αφορούν τον παίχτη.

- Tiles: Περιέχει κλάσεις που υλοποιούν τις λειτουργίες των πλακιδίων του παιχνιδιού.
- Round: Περιέχει την κλάση Round που χειρίζεται τις λειτουργίες των γύρων του παιχνιδιού.

1. Το πακέτο Board

Το πακέτο Board περιέχει την κλάση Board η οποία υλοποιεί τις λειτουργίες του ταμπλό του παιχνιδιού και εξής χαρακτηριστικά:

- `protected ArrayList<Tile> Mosaic_Area;`
- `protected ArrayList<Tile> Amphora_Area;`
- `protected ArrayList<Tile> Skeleton_Area;`
- `protected ArrayList<Tile> Statue_Area;`
- `protected ArrayList<Tile>Entrance_Area;`

==> Το κάθε ArrayList αναφέρεται στις 5 διαφορετικές περιοχές του ταμπλό όπου τοποθετούνται τα αντίστοιχα πλακίδια.

Οι μέθοδοι της κλάσης Board είναι οι εξής:

- `public Board() // κατασκευάστρια μέθοδος`
- `public void Place_Tile(Tile tile)// μέθοδος για την προσθήκη ενός πλακιδίου στο arraylist που ταιριάζει με τον τύπο του`
- `public ArrayList<Tile> get_Area(String choice)// μέθοδος για την επιστροφή του arraylist καθορισμένου απο την επιλογή (choice)`

2. Το πακέτο Cards

Το πακέτο Cards περιέχει την αφηρημένη κλάση Character της οποίας στόχος είναι η υλοποίηση των λειτουργιών των καρτών χαρακτήρων και έχει τα εξής χαρακτηριστικά

- `protected boolean Used;` // πεδίο της κλάσης που υποδηλώνει αν το skill του χαρακτήρα έχει χρησιμοποιηθεί ή όχι.
- `protected Player Owner;`// πεδίο της κλάσης που αναφέρεται στο ποίος παίκτης είναι ο ιδιοκτήτης της συγκεκριμένης κάρτας.
- `protected String Type;`// πεδίο της κλάσης που αναφέρεται στον τύπο του χαρακτήρα(digger,assistant etc.)

Οι μέθοδοι της κλάσης:

- `Character(String type,Player p)` // κατασκευάστρια μέθοδος με παράμετρο ένα αντικείμενο της κλάσης player,η μέθοδος δημιουργεί μια κάρτα χαρακτήρα τύπου type με ιδιοκτήτη έναν παίκτη.
- `public abstract void Character_Skill()`// αφηρημένη μέθοδος η οποία εφαρμόζει το skill μιας κάρτας χαρακτήρα.
- `public abstract Player Get_Owner()`// αφηρημένη μέθοδος η οποία επιστρέφει τον ιδιοκτήτη της κάρτας.
- `public abstract void Set_Owner(Player p)`// αφηρημένη μέθοδος η οποία εκχωρεί έναν ιδιοκτήτη στην συγκεκριμένη καρτα.
- `public abstract String getType()`//επιστρέφει τον τύπο του χαρακτήρα σε μορφή string
- `public abstract boolean is_Used()`// επιστρέφει true αν ο χαρακτήρας έχει χρησιμοποιηθεί αλλιώς επιστρέφειfalse.

- `public abstract void setUsed(boolean value)// Αναθέτει την τιμή value στο πεδίο Used`
- `public abstract String get_skill()// Επιστρέφει την περιγραφή του skill του χαρακτήρα`
- `public abstract boolean has_completed_skill()// Αν ο χαρακτήρας έχει ολοκληρώσει το skill του επιστρέφει true αλλιώς επιστρέφει false`

Η κλάση Character επεκτείνεται από τις παρακάτω κλάσεις:

- Professor
- Digger
- Archaeologist
- Assistant

Οι οποίες υλοποιούν τις μεθόδους της κλάσης Character και η κάθε μια έχει την δική της κατασκευάστρια μέθοδο όπου χρησιμοποιώντας την κατασκευάστρια μέθοδος της Character δημιουργούν και την αντίστοιχη κάρτα χαρακτήρα.

Η πιό σημαντική λειτουργία των επεκτάσεων της Character είναι η χρήση του skill που υλοποιείται από την μέθοδο Character_Skill η οποία είναι overriden.

Ο κάθε χαρακτήρας διαφέρει στο πως αξιοποιεί την μέθοδο.Οι διαφορές είναι στους περιορισμούς των πλακιδίων που επιτρέπει ο χαρακτήρας στον παίκτη να τραβήξει από το ταμπλό.

Στον πυρήνα τους όμως οι Character_Skill για κάθε κλάση χαρακτήρα ακολουθούν την ίδια μεθοδολογία.

Αρχικά οι μέθοδοι δέχονται ως παραμέτρους την ενεργώ εκδοχή της κλάσης GUI και ένα Action event που αναφέρεται σε κλικ ενός πλακιδίου.

Στη συνέχεια αρχικοποιούνται μεταβλητές απαραίτητες για να ξέρουμε πιο πλακίδιο έχει επιλεχτεί και απο ποία περιοχή.

Ύστερα αφού γνωρίζουμε όλες τις πληροφορίες που χρειαζόμαστε χρησιμοποιούμε μεθόδους της κλάσης gui που θα αναλυθούν αργότερα για να λάβει ο παίκτης στην κατοχή του τα πλακίδια που έχει επιλέξει ενώ ταυτόχρονα ελέγχεται το αν οι ενέργειες του παίκτη γίνονται στα πλαίσια του skill του χαρακτήρα ,αν όχι εμφανίζεται μήνυμα λάθους και η ενέργεια δεν ολοκληρώνεται.

3. Το πακέτο Player

Το πακέτο Player περιέχει την κλάση Player η οποία διαχειρίζεται τις λειτουργίες και τις ενέργειες του παίκτη και έχει τα παρακάτω χαρακτηριστικά:

- `private String Player_Id;// Το αναγνωριστικό του παίκτη`
- `private ArrayList<FindingTiles> Owned_Tiles;// Array list που περιέχει finding tiles,σκοπός είναι η αποθήκευση των πλακιδίων του παίκτη`
- `private ArrayList<Character>CharacterList;// Arraylist που περιέχει τις κάρτες χαρακτήρων του παίκτη`
- `private String Color;// Το αναγνωριστικό χρώμα του παίκτη`

- `private int Points;// Το ατομικό σκορ του παίκτη`
- `private int tiles_taken;//Τα πλακίδια που έχει πάρει ο παίκτης απ το ταμπλό,η τιμή ανανεώνεται σε κάθε γύρο`
- `private boolean has_drawn;//Ελέγχει αν ο παίκτης έχει τραβήξει πλακίδια απ'τη σακούλα`
- `private JPanel chosen_area;//Η περιοχή απ την οποία ο παίκτης πήρε πλακίδια`
- `private int sphinxes;//Αριθμός απο σφιγγες που εχει συγκεντρώσει ο παίκτης,θα χρησιμοποιηθεί στην καταμέτρηση του σκορ`
- `private int caryatids;//Αριθμός απο καρυατιδες που εχει συγκεντρώσει ο παίκτης,-//-`

Οι μέθοδοι της κλάσης:

- `public Player()// Κατασκευάστρια μέθοδος η οποία χρησιμοποιείται σε περίπτωση που το παιχνίδι είναι με έναν παίκτη για να δημιουργήσει τον grave_robbers`
- `public Player(String id,Colors color)// Κατασκευάστρια μέθοδος η οποία δημιουργεί έναν καινούργιο παίκτη με χαρακτηριστικό id και χρώμα με τιμή από την enumeration class Colors και αρχικοποιεί τα υπόλοιπα πεδία με default τιμές`
- `public JPanel getChosen_area()// μέθοδος που επιστρέφει το τωρινό πεδίο chosen_area`
- `public JPanel setChosen_area(JPanel chosen_area)// μέθοδος που αναθέτει τιμή στο πεδίο chosen_area`
- `public void setPlayer_Id(String id)// μέθοδος που εισάγει ένα καινούργιο χαρακτηριστικό id σε παίκτη`

- public String getPlayer_Id()// μέθοδος που επιστρέφει το χαρακτηριστικό του παίκτη
- public int getPoints()// μέθοδος που επιστρέφει το σκορ του παίκτη
- public void setPoints(int p)//μέθοδος που εισάγει ένα νέο σκορ p στον παίκτη αφού γίνουν οι απαραίτητοι έλεγχοι εγκυρότητας του καινούργιου σκορ
- public void Assign_Tiles(FindingTiles tile)// μέθοδος που παίρνει ως όρισμα ένα πλακίδιο και το εισάγει στον πίνακα πλακιδίων που ανήκουν στον παίκτη,η εισαγωγή στον πίνακα γίνεται ταξινομημένα ώστε τα πλακίδια να είναι ομαδοποιημένα ανάλογα με τον τύπο τους
- public void use_character(Character c)// μέθοδος που επιτρέπει στον παίκτη να χρησιμοποιήσει το skill μιας κάρτας χαρακτήρα που ορίζεται από την παράμετρο c
- public ArrayList<ArrayList<FindingTiles>> getOwned_Tiles()// μέθοδος που επιστρέφει το arraylist of arraylists των πλακιδίων που ανήκουν στον παίκτη
- public void setTiles_taken(int num)// Μέθοδος που αναθετει τιμη στο πεδιο tiles taken
- public void Increment_tiles_taken()// Μεθοδος που αυξανει το πεδιο tiles taken(tiles taken++) σε καθε κληση
- public int getTiles_taken()// Επιστρεφει την τιμη του πεδιου tiles taken
- public boolean hasdrawn()// Επιστρεφει την τιμη του πεδιου has_drawn
- public void setHas_drawn(boolean value)// Αναθετει την τιμη value στο πεδιο has_drawn
- public String get_color()// Επιστρεφει το χρωμα του παικτη

- `public Character get_char_by_type(String type)// Διασχιζει την λιστα των καρτων χαρακτηρων του παικτη και επιστρεφει αυτον που ο τυπος του ταιριαζει με την παραμετρο type`
- `public void add_characters(Character c)// Εισαγει ενα αντικειμενο τυπου Character στο πεδιο Character_list`
- `Τελος εχουν υλοποιηθει setters και getters για τα πεδια sphinxes και caryatids`

4. Το πακέτο Round

Το πακέτο Round περιέχει την κλάση Round η οποία διαχειρίζεται τις λειτουργίες και τις ενέργειες που πραγματοποιούνται κατά την διάρκεια ενός γύρου του παιχνιδιού και έχει τα παρακάτω χαρακτηριστικά:

- `private Player current;//Το πεδιο αφορά τον παίκτη του οποίου είναι η σειρά να παίξει`
- `private int turn;//Το index του γύρου που πραγματοποιείται εκείνη την στιγμή`
- `private Arraylist<Player>order;//Το πεδιο αποθηκευει αντικειμενα τυπου player συμφωνα με την σειρα με την οποια παιζουν οι παικτες`

Οι μέθοδοι της κλάσης:

- `public Round()//Κατασκευάστρια μέθοδος που δημιουργεί έναν νέο γύρο με default τιμές ,συγκεκριμένα το round index θα είναι 0`
- `public Round(int turn,Player p,boolean ready)//Κατασκευάστρια μέθοδος που δημιουργεί έναν νέο γύρο του οποίου το index θα καθορίζεται από την παράμετρο turn αφού γίνουν οι κατάλληλοι έλεγχοι εγκυρότητας,ο παίκτης του οποίου είναι η σειρά`

καθορίζεται από την παράμετρο p και τέλος το αν είναι έτοιμος να παίξει καθορίζεται από την παράμετρο ready

- `public void setTurn(int turn)`//μέθοδος που ενημερώνει το index του γύρου σύμφωνα με την παράμετρο turn αφού γίνουν οι απαραίτητοι έλεγχοι εγκυρότητας.Επισης υπολογιζεται το ποιανου παικτη ειναι σειρα χρησιμοποιοντας τον πινακα order,ο πρωτος γυρος αρχικοποιειται με τιμη 0 οποτε στην κληση της μεθοδου ο γυρος θα χει τιμη >=0 , ετσι σε καθε κληση κανουμε turn mod το μεγεθος του order για να βρουμε το index του παικτη του οποιου ειναι η σειρα να παιξει.

- `public int getTurn()`//μέθοδος που επιστρέφει το τωρινό index του γύρου

- `public void set_order(ArrayList<Player>order)`//μέθοδος που δεχεται ως παραμετρο μια λιστα με αντικειμενα τυπου Player και αντιγραφει την παραμετρο στο πεδιο order,η μεθοδος χρησιμοποιειται για να καθορισει τον πρωτο παικτη στη σειρα και να αρχισει ο γυρος του

- `public void getCurrent()`//Επιστρεφει τον παικτη του οποιου ειναι η σειρα

- `public ArrayList<Player> getOrder()`//Επιστρεφει την λιστα της σειρας

5. Το πακέτο Tile

Το πακέτο Tile περιέχει κλάσεις οι οποίες διαχειρίζονται τις λειτουργίες των πλακιδίων του παιχνιδιού και είναι οι εξής:

- Tile

- TileContainer
- Colors
- FindingTiles
- LandslideTiles
- MosaicTile
- SkeletonTile
- AmphoraTile
- StatueTile
- CaryatidTile
- SphynxTile

- Το interface Tile αφορά όλα τα πλακίδια του παιχνιδιού και περιέχει τις εξής μεθόδους:

```
void place_Tile(Tile t);//μέθοδος για την τοποθέτηση ενός tile t στην περιοχή του board που ταιριάζει στον τύπο του.
String toString();//μέθοδος που επιστρέφει τον τύπο του tile σε string format.
```

- **<u>Το interface Tile υλοποιείται από τις εξής κλάσεις</u>**
  - Την κλάση FindingTiles η οποία αφορά τα πλακίδια που μπορούν να βρεθούν και συλλεχθούν από τους παίκτες και έχεις τα εξής χαρακτηριστικά:

- `Player owner;//Το πεδίο αφορά τον ιδιοκτήτη του πλακιδίου δηλαδή έναν παίκτη`

- `String type;//Το πεδίο αφορά τον τύπο του πλακιδίου,αν είναι αμφορέας ,σκελετός,άγαλμα.`

Οι μέθοδοι της κλάσης:

- `public FindingTiles(String type)//Κατασκευάστρια μέθοδος που δημιουργεί ένα καινούργιο πλακίδιο με τύπο καθορισμένο από την παράμετρο type`

- `public FindingTiles(String type,Player p)//Κατασκευάστρια μέθοδος που δημιουργεί ένα καινούργιο πλακίδιο με τύπο καθορισμένο από την παράμετρο type και ιδιοκτήτη καθορισμένο από την παράμετρο p`

- `public Player getOwner()//Μέθοδος που επιστρέφει τον ιδιοκτήτη του πλακιδίου`

- `public void setOwner(Player player)//Μέθοδος που αναθέτει στο πλακίδιο έναν ιδιοκτήτη`

- `public void place_Tile(Tile t)//Μέθοδος που υπερφορτώνει την μέθοδο place_tile του interface`

➢ **Η κλάση FindingTiles αποτελεί γονική κλάση των κλάσεων:**

➢ AmpohraTile η οποία διαχειρίζεται τα πλακίδια αμφορέων και εχει τα εξής χαρακτηριστικά:

- `protected Colors color;//Το πεδίο αφορά το χρώμα του αμφορέα και είναι τύπου color`

- `protected ArrayList<AmphoraTile>Amphor`

as;//Array list με σκοπό την ομαδοποίηση αμφορέων

Οι μέθοδοι της κλάσης:

- public AmphoraTile(Colors color)//κατασκευάστρια μέθοδος που δημιουργεί έναν νέο αμφορέα με χρώμα καθορισμένο από την παράμετρο color
- public AmphoraTile(Colors color, Player p)//κατασκευάστρια μέθοδος που δημιουργεί έναν νέο αμφορέα με χρώμα καθορισμένο από την παράμετρο color και ιδιοκτήτη έναν παίκτη καθορισμένο από την παράμετρο p
- public ArrayList<AmphoraTile> getAmphoras()//μέθοδος που επιστρέφει τον arraylist με τους αμφορείς
- public void setAmphoras(AmphoraTile t)//μέθοδος που εισάγει έναν αμφορέα t στην λίστα των αμφορέων

➢ MosaicTile η οποία είναι υπεύθυνη για την διαχείριση πλακιδίων τύπου μωσαικού και έχει τα εξής χαρακτηριστικά:

- protected Colors color;//Το πεδίο αφορά το χρώμα του μωσαικού
- protected ArrayList<MosaicTile>Mosaics//Arraylist για την ομαδοποίηση των μωσαικών

Οι μέθοδοι της κλάσης:

- public MosaicTile(Colors color)//κατασκευάστρια μέθοδος που δημιουργεί ένα νέο πλακίδιο μωσαικού με χρώμα καθορισμένο από την παράμετρο color και κάνει χρήση της κατασκευάστριας μεθόδου της γονικής κλάσης
- public MosaicTile(Colors color, Player p)//κατασκευάστρια μέθοδος που δημιουργεί ένα νέο πλακίδιο μωσαικού με χρώμα καθορισμένο από την παράμετρο color και ιδιοκτήτη έναν παίκτη καθορισμένο από την παράμετρο p, κάνει χρήση της κατασκευάστριας μεθόδου της γονικής κλάσης
- public void setMosaics(MosaicTile t)//μέθοδος που προσθέτει ένα πλακίδιο στον πίνακα μωσαικών ,η εκχώρηση πρέπει να είναι τέτοια ώστε να διατηρείται η ομαδοποίηση των μωσαικών
- public ArrayList<MosaicTile> getMosaics()//μέθοδος που επιστρέφει τον πίνακα των μωσαικών

➢ SkeltetonTile ,η κλάση SkeletonTile υλοποιεί τις λειτουργίες και τις ενέργειες των πλακιδίων σκελετών και έχει τα εξής χαρακτηριστικά:

- protected String type;//Το πεδίο αναφέρεται στον τύπο του σκελετού ,αν είναι ενήλικας ή παιδί

- protected ArrayList<SkeletonTile> Skeleton_Family =new ArrayList<>();//Σκοπός του arraylist είναι η αποθήκευση ολοκληρωμένων οικογενειών σκελετών
- protected ArrayList<SkeletonTile>Skeleton_Parts=new ArrayList<>();//Arraylist για την αποθήκευση των κομματιών των σκελετών
- protected boolean isComplete;//Το πεδίο καθορίζει αν όλα τα κομμάτια του σκελετού έχουν συμπληρωθεί

Οι μέθοδοι της κλάσης:

- public SkeletonTile()//Κατασκευάστρια μέθοδος που δημιουργεί ένα νέο πλακίδιο σκελετού με default τιμές
- public SkeletonTile(String type)//Κατασκευάστρια μέθοδος που δημιουργεί ένα νέο πλακίδιο σκελετού με τύπο(ενήλικας,παιδί) καθορισμένο από την παράμετρο type
- public String get_type()//μέθοδος που επιστρέφει αν ένας σκελετός είναι ενήλικας ή παιδί
- public void setType(String type)//μέθοδος που αναθέτει στον σκελετό έναν τύπο

- `public void setSkeleton_Family(StatueTile skeleton)`//μέθοδος που εισάγει ένα πλακίδιο τύπου σκελετού στην οικογένεια σκελετών
- `public boolean is_Complete()`//Μέθοδος που επιστρέφει true αν ο σκελετός έχει όλα του τα κομμάτια και false αν όχι

- Colors,η κλάση απαρίθμησης Colors περιέχει στα χαρακτηριστικά της όλα τα χρώματα που μπορούν να πάρουν τα πλακίδια που έχουν χρώμα,τα χρώματα είναι,RED,BLUE,BROWN,GREEN, ORANGE,YELLOW,PURPLE
- Η κλάση StatueTile η οποία επεκτείνεται από τις κλάσεις SphynxTile και CaryatidTile και είναι υπεύθυνες για τις λειτουργίες σε πλακίδια αγαλμάτων,Τα χρακτηριστικά της κλάσης είναι τα εξής:
  - `protected ArrayList<SphynxTile> Sphynxes;`//Arraylist που αποθηκεύει αγάλματα τύπου sphynx

- protected ArrayList<CaryatidTile>Caryatids;//Arraylist που αποθηκεύει αγάλματα τύπου Caryatid

Οι μέθοδοι της κλάσης:

- public StatueTile(String type)//Κατασκευάστρια μέθοδος που δημιουργεί ένα νέο άγαλμα με τύπο καθορισμένο από την παράμετρο type
- public StatueTile(String type, Player p)//Κατασκευάστρια μέθοδος που δημιουργεί ένα νέο άγαλμα με τύπο καθορισμένο από την παράμετρο type και ιδιοκτήτη καθορισμένο από την παράμετρο p
- public void Set_Sphynx(SphynxTile t)//μέθοδος που εισάγει ένα πλακίδιο τυπου sphynx στον πίνακα με τα αγάλματα αντίστοιχου τύπου
- public void Set_Caryatid(CaryatidTile t)//μέθοδος που εισάγει ένα πλακίδιο τυπου caryatid στον πίνακα με τα αγάλματα αντίστοιχου τύπου
- public ArrayList<SphynxTile> get_Sphynx()//μέθοδος που

επιστρέφει τον πίνακα με πλακίδια τύπου sphynx

- public ArrayList<CaryatidTile> get_Caryatids()//μέθοδος που επιστρέφει τον πίνακα με πλακίδια τύπου caryatid

➢ Οι κλάσεις SphynxTile και CaryatidTile δεν έχουν attributes παρά μόνο κατασκευάστριες μεθόδους που κάνουν χρήση των κατασκευαστριών μεθόδων τής γονικής κλάσης δίνοντας τις κατάλληλες παραμέτρους

- Η κλάση LandslideTile αφορά τα tiles τα οποία τοποθετούνται στην είσοδο ,η κλάση δεν έχει attributes και υπερφορτώνει τις μεθόδους του interface Tile

- Η Κλάση TileContainer υλοποιεί της λειτουργίες της τσάντας που περιέχει τα πλακίδια από την οποία τραβάνε οι παίκτες και έχει τα εξής χαρακτηριστικά:

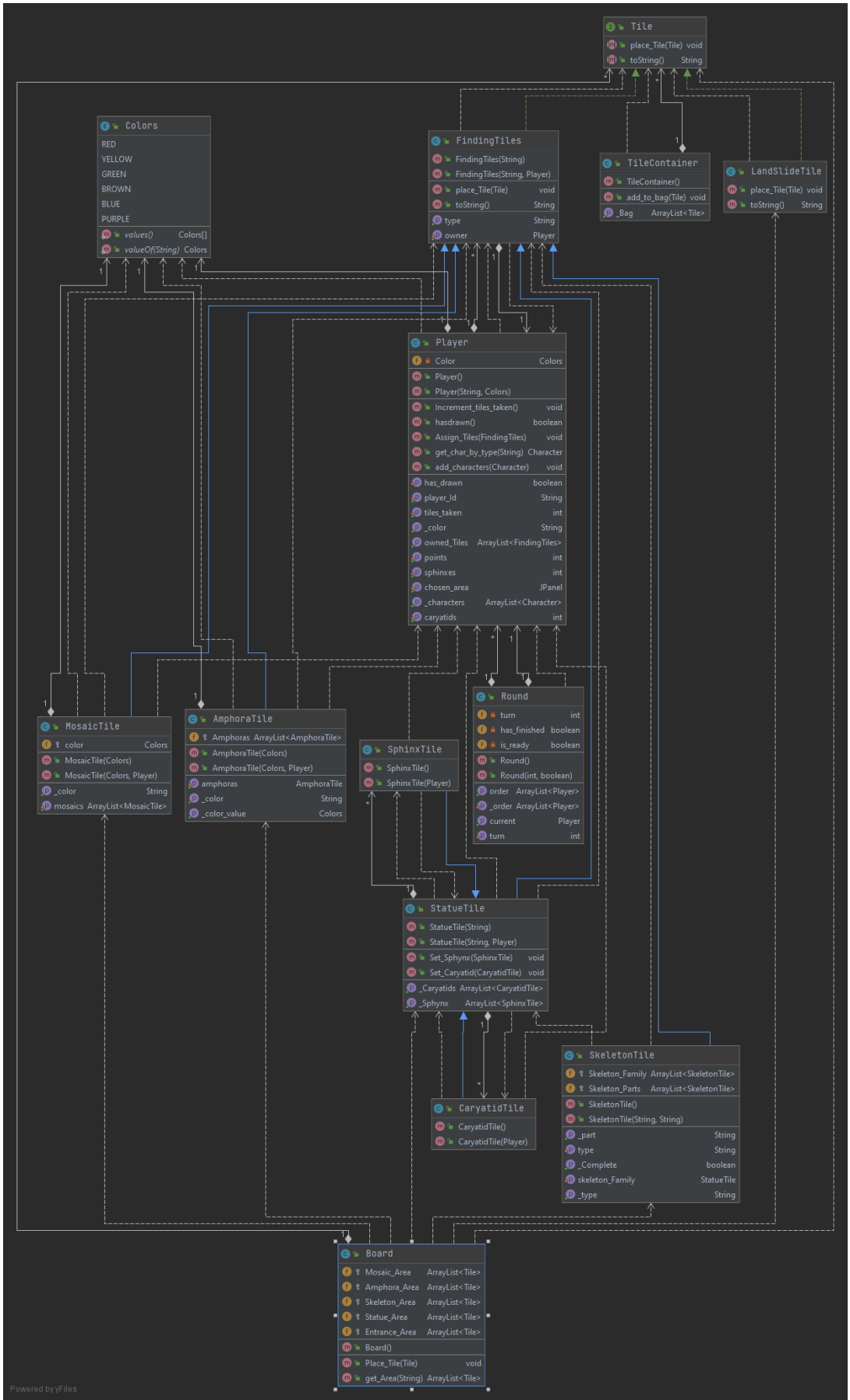
  - ArrayList<Tile> Bag;//Arraylist που περιέχει πλακίδια προσομοιώνοντας την τσάντα με τα πλακίδια

Οι μέθοδοι της κλάσης:

- public TileContainer()//Κατασκευάστρια Μέθοδος που δημιουργεί μια καινούργια τσάντα με πλακίδια κάνοντας την απαραίτητη διαχείριση μνήμης στο arraylist
- public void add_to_bag(Tile tile)//Μέθοδος που εισάγει ένα πλακίδιο στην τσάντα

Δημήτρης Βλάχος

UML ΔΙΆΓΡΑΜΜΑ ΓΙΑ ΤΟ ΠΑΚΈΤΟ MODEL



6. Το πακέτο Controller

Το πακέτο Controller περιέχει την γονική κλάση Controller και την υποκλάση αυτής ,την Game_Controller.

- Η κλάση Controller έχει τα εξής χαρακτηριστικά

  - `protected Round round;//Μεταβλητή τύπου round που αποθηκεύει τον τωρινό γύρο του παιχνιδιού`
  - `rotected Player who_plays;//μεταβλητή τύπου player που καθορίζει το ποιός παίκτης παίζει την παρούσα στιγμή`
  - `protected ArrayList<Player>Players;//πίνακας που αποθηκεύει τους παίκτες που είναι ενεργοί`
  - `protected TileContainer bag;//Η τσαντα η οποια περιεχει ολα τα πλακιδια`
  - `protected int num_of_Players;//Αριθμος ενεργων παικτων`
  - `private Game_Controller game;//Αντικειμενο τυπου game controller`
  - `private boolean single_player;//Ελεγχει αν το παιχνίδι θα παιχτει απο εναν παιχτη`
  - `protected Board board;//Αντικειμενο της κλασης Board η οποια θα αρχικοποιηθει στην κλαση controller`

  Οι μέθοδοι της κλάσης:

- public Controller(ArrayList<String>ids,ArrayList<String> colors)//Κατασκευάστρια μέθοδος η οποία δημιουργεί ένα καινούργιο παιχνίδι σύμφωνα με τις παραμέτρους που δέχεται,αν ο αριθμός των παικτών ειναι 1 ξεκινάει το παιχνίδι σε singleplayer mode.
- public static void main(String[]args)//η main μέθοδος ,απο αυτήν αρχίζει το παιχνίδι αφού δημιουργεί ένα instance της controler .Υπαρχει για λόγους debugging
- public void game_start(Game_Controller game)//Μέθοδος που ξεκινάει το παιχνίδι αφού λαμβάνει ως παράμετρο ένα instance της κλάσης game_controller μέσω αυτού να προετοιμάσει το παιχνίδι για εκκίνηση
- public ArrayList<Player> getPlayers()//μέθοδος που επιστρέφει τον πίνακα με τους ενεργούς παίκτες

➢ Η κλάση Game_controller επεκτείνει την κλάση Controller ,είναι υπεύθυνη για το περιβάλλον του παιχνιδιού,τις ενέργειες του menu του παιχνιδιού και της γενικής του κατάστασης ,έχει τα εξής χαρακτηριστικά:

- protected boolean game_state;//Το πεδίο αποθηκεύει την παρούσα κατάσταση του παιχνιδιού ,true αν είναι ενεργό false αν όχι
- protected GUI gui;//Αποθηκεύει την εκδοχή της GUI που θα χρησιμοποιηθεί

- private Player grave_robber;//Αντικείμενο τύπου Player που θα χρησιμοποιηθεί για τις λειτουργίες του grave robber
- private boolean single_player;//Παίρνει τιμή true αν το παιχνίδι είναι σε single player mode αλλιώς παίρνει τιμή false

Οι μέθοδοι της κλάσης:

- public Game_Controller()//Κατασκευάστρια μέθοδος που δημιουργεί ένα νέο game controller θέτοντας την default τιμή false στο πεδίο game state
- public void setGame_state(boolean state) //μέθοδος που ενημερώνει το πεδίο game_state σύμφωνα με την παράμετρο state
- public boolean getGame_state()//μέθοδος που επιστρέφει την παρούσα κατάσταση του παιχνιδιού
- public void initialize_enviroment(ArrayList <Player> Players, Round round,boolean mode)//μέθοδος που θέτει σε λειτουργία το γραφικό περιβάλλον ώστε να αρχίσει το παιχνίδι.Δέχεται ως παραμέτρους μια λίστα που περιέχει τους ενεργούς παίκτες ,ένα αντικείμενο round και το mode το οποίο είναι true αν το παιχνίδι είναι σε singleplayer mode και false αν όχι.Στη συνέχεια καλείται η συνάρτηση generate tiles

η οποία δημιουργεί όλα τα αντικείμενα για τα tiles που θα χρησιμοποιηθούν και τοποθετούνται στην τσάντα.Ύστερα τοποθετούνται τα αρχικά tiles στο ταμπλό σύμφωνα με τους κανόνες του παιχνιδιού.Αν το παιχνίδι είναι σε single_player mode τότε δημιουργείται ο grave robber και τοποθετούνται στην είσοδο 8 πλακίδια κατολίσθησης.

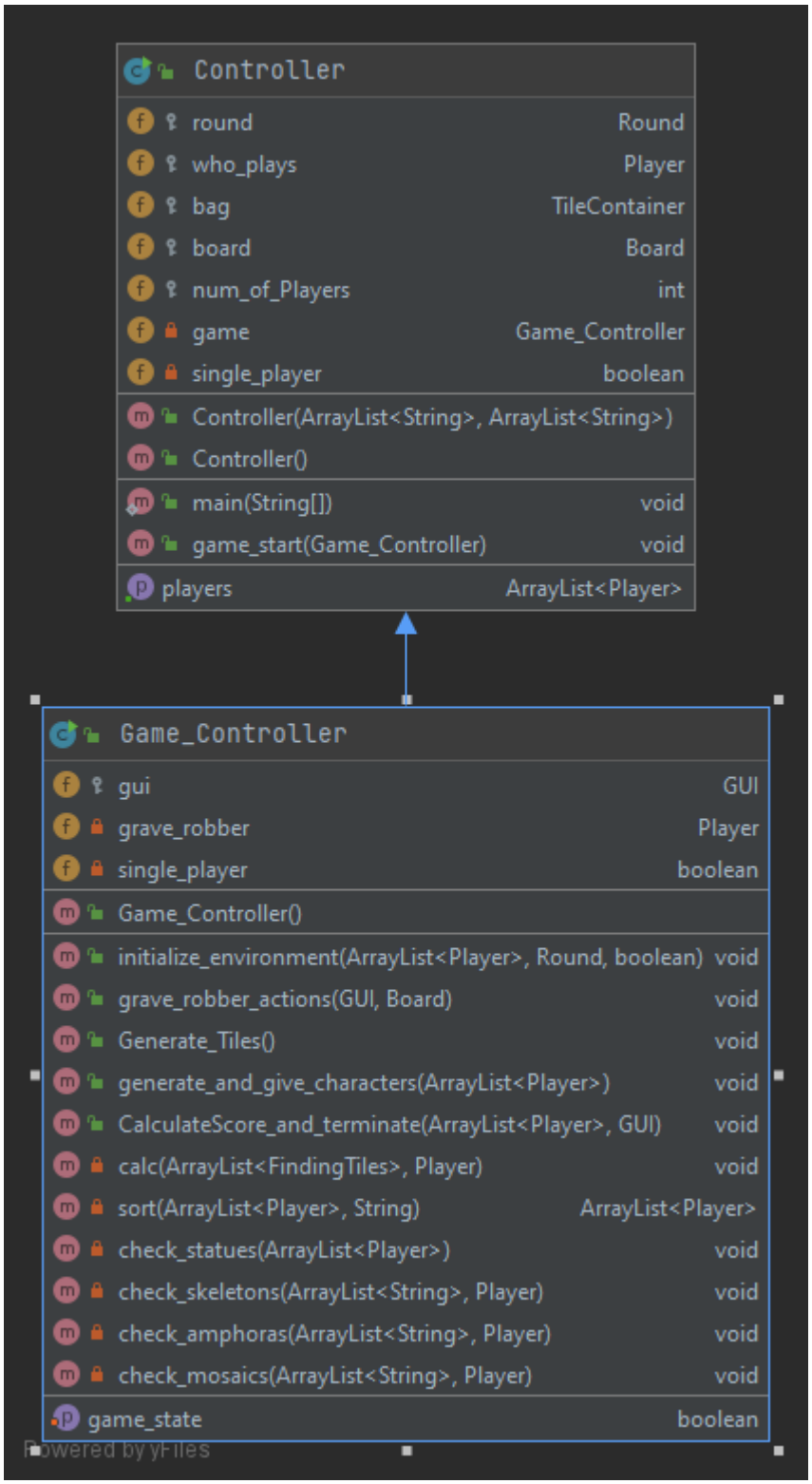
- public void give_and_give_character(ArrayList<Player>Players)//μέθοδος που δημιουργεί μοιράζει κάρτες χαρακτήρων στους παίκτες μόλις αρχίσει το παιχνίδι
- public void grave_robber_actions(GUI gui,Board board)//Η μεθοδος υλοποιει τις ενεργεις του grave robber , μεσω της ενεργου εκδοχης της κλασης gui ελεγχει ποσα και ποια πλακιδια ειναι τοποθετημενα στο ταμπλο ,δημιουργει αντικειμενα τυπου tile και για την μετατροπη του jbutton σε tile χρησιμοποιει το hash map button mappers της gui,τελος διαγραφει τα πλακιδια απο το ταμπλο και τα εισαγει στα owned tiles του grave robber
- public void Generate_Tiles()//Δημιουργει αντικειμνα τυπου Tile συμφωνα με τους κανονες του παιχνιδιου οριζοντας τα καταλληλα attributes,στην συνεχεια τα τοποθετει στην τσαντα

Μια πολύ σημαντική λειτουργία της κλάσης Game_Controller είναι ο υπολογισμός του

σκορ,αυτό επιτυγχάνεται μέσω των συναρτήσεων:

- public void CalculateScore_and_terminate(ArrayList<Player>active_players,GUI gui)//Η μεθοδος καλειται απο την κλαση gui και παιρνει ως παραμετρο την εκδοχη της κλασης και την λιστα με τους ενεργους παικτες,αν το παιχνιδι ειναι single player βαζει στην λιστα των παικτων με την οποια θα γινει ο υπολογισμος του σκορ τον grave robber.Στην συνεχεια καλειται η συναρτηση calc η οποια ομαδοποιει τα owned tiles του καθε παικτη σε λιστες απο strings ωστε η επεξεργασια των δεδομενων να ειναι πιο ευκολη,με τη σειρα της η calc καλει τις συναρτησεις που ειναι υπευθηνες για τον υπολογισμο των ποντων του καθε τυπου πλακιδιων.Τελος δημιουργειται ενας πινακας με τα σκορ σε φθινουσα διαταξη και εμφανιζεται σε ενα παραθυρο ,μετα το παιχνίδι τερματιζεται.

UML ΔΙΆΓΡΑΜΜΑ ΤΟΥ ΠΑΚΈΤΟΥ CONTROLLER


Controller
round Round
who_plays Player
bag TileContainer
board Board
num_of_Players int
game Game_Controller
single_player boolean
Controller(ArrayList<String>, ArrayList<String>)
Controller()
main(String[]) void
game_start(Game_Controller) void
players ArrayList<Player>
Game_Controller
gui GUI
grave_robber Player
single_player boolean
Game_Controller()
initialize_environment(ArrayList<Player>, Round, boolean) void
grave_robber_actions(GUI, Board) void
Generate_Tiles() void
generate_and_give_characters(ArrayList<Player>) void
CalculateScore_and_terminate(ArrayList<Player>, GUI) void
calc(ArrayList<FindingTiles>, Player) void
sort(ArrayList<Player>, String) ArrayList<Player>
check_statues(ArrayList<Player>) void
check_skeletons(ArrayList<String>, Player) void
check_amphoras(ArrayList<String>, Player) void
check_mosaics(ArrayList<String>, Player) void
game_state boolean
Powered by yFiles

6. Το πακέτο View

Το πακέτο View περιέχει την κλάση GUI και την κλάση Main_menu που υλοποιεί το γραφικό περιβάλλον του παιχνιδιού.

Η ΚΛΑΣΗ GUI:

Τα χαρακτηριστικά της κλάσης αφορούν τις αρχικοποιήσεις των κουμπιών του γραφικού περιβάλλοντος και των μενού

Οι μέθοδοι της κλάσης:

- `public GUI(String []Player_ids)` //Κατασκευάστρια μέθοδος που δημιουργεί ένα νέο γραφικό περιβάλλον ,το όρισμα είναι ένας πίνακας με τα αναγνωριστικά των παικτών τα οποία θα πρέπει να φαίνονται στο γραφικό περιβάλλον
- `public void Initialize_Environment()` //μέθοδος που δημιουργεί τα components του γραφικού περιβάλλοντος ,όπως τις εικόνες ,τα containers ,τα μενού κτλ
- `private void set_Area_Layouts()` //Δημιουργει τις περιοχες στις οποιες θα τοποθετουνται τα πλακιδια,πιο συγκεκριμένα οι περιοχες ειναι Jpanels και το layout που χρησιμοποιειται για αυτες ειναι το grid layout .
- `public void Place_Tile(Tile tile)` //Η μεθοδος ειναι υπευθυνη για την δημιουργια ενος πλακιδιου και την τοποθετηση του πανω στην καταλληλη περιοχη στο ταμπλο.Για να επιτευχθει αυτο αρχικα μεσω των βοηθητικων συναρτησεων της tile και των υλοποιησεων της κατασκευαζεται το ονομα του αρχειου που

περιεχει την εικονα η οποια συναυδει με το πλακιδιο.Υστερα καλειται η συναρτηση create_tile_button η οποια δημιουργει το κουμπι/πλακιδιο με την εικονα ,του εισαγει τον καταλληλο action listenerκαι action command.Τελος ανανεωνονται τα πεδια του board με τα υπαρχοντα πλακιδια πανω στο ταμπλο και το πλακιδιο που αντιστοιχει στο κουμπι εισαγεται στο button mappers με αναγνωριστικο ενα string με το ονομα του.

- public void set_player_colors()//Η συναρτηση ειναι υπευθηνη για την αναθεση χρωματων στους παικτες.Πιο συγκεκριμενα ,το πεδιο color καθε παικτη ειναι τυπου Colors,ομως για την γραφικη διεπαφη πρεπει να “μεταφρασουμε” αυτην την τιμη σε χρωμα το οποιο μπορει να αναπαρασταθει με γραφικα.Ετσι με την κληση της συναρτησης get_color_by_name πετυχαινουμε ακριβως αυτο.
- public void Update_player_tiles(Tile t)//Εισαγει ενα πλακιδιο στη λιστα των κατεχομενων πλακιδιων του παικτη που ειναι ο γυρος του.
- private void grave_rob()//Καλει την μεθοδου του game controller που υλοποιει τον grave robber
- private void finish_game()//Καλειται μολις η εισοδος γεμισει με 16 πλακιδια ,καλει την μεθοδο του game controller που υπολογιζει τα σκορ και τερματιζει το παιχνιδι.
- private void use_skill(Character c,ActionEvent event)//Καλει την μεθοδο που υλοποιει το σκιλ του συγκεκριμενου χαρακτηρα
- public void Update_owned_tiles_panel(JPanel card,Container parent,JButton clicked_button)//Οταν ο παικτης παρει ενα πλακιδιο αυτο πρεπει να τοποθετηθει στο panel

card(card layout) του παικτη ,για να γινει αυτο πρεπει το πλακιδιο να διαγραφει απο την περιοχη στην οποια βρισκοταν και το panel του παικτη να επανοριστει με το καινουργιο component που θα περιεχει(το προς τοποθετηση πλακιδιο).Επισης διαγραφεται ο action listener απο το πλακιδιο-κουμπι διοτι πλεον δεν εχει υφισταται.

- public void Initialize_Buttons()//μέθοδος που δημιουργεί τα κουμπιά

Οι εμφωλευμένες κλάσεις:

- Οι εμφωλευμένες κλάσεις υλοποιούν

τις διάφορες ενέργειες που γίνονται στο γραφικό περιβάλλον του παιχνιδιού

- private class use_character implements ActionListener//η κλάση υλοποιεί την λειτουργία της χρήσης μιας κάρτας χαρακτήρα απο τον παίκτη,για να γινε αυτο ο χαρακτηρας αποθηκευεται σε μια μεταβλητη current χρησιμοποιοντας το action command του κουμπιου που μας πληροφορει για τον τυπο του το οποιο στελνεται ως παραμετρος στην get_character_by_type.Αν ο χαρακτηρας εχει χρησιμοποιηθει εμφανιζεται μηνυμα λαθους διαφορετικα η μεταβλητη character mode γινεται true.Η character mode χρησιμοποιειται για να γινεται ελεγχος αν ο παικτης παιρνει πλακιδια συμφωνα με την default κινηση του (2 πλακιδια απο μια περιοχη ανα γυρο) η γινεται συμφωνα με το αντιστοιχο character skill.

- private class take_tiles implements ActionListener//η κλάση υλοποιεί την λειτουργία της περισυλλογής πλακιδίων απο το ταμπλο.Εχει δυο λειτουργιες ,το default mode και το character mode οι οποιες υλοποιουνται απο τις συναρτησεις με τα αντιστοιχα ονοματα.

- Αν ο παικτης εχει επιλεξει να χρησιμοποιησει χαρακτηρα τοτε το πεδιο της gui character mode

είναι true αρα καλειται η συναρτηση character mode η οποια με την σειρα της καλει την use_skill .Στην συνεχεια μερικα flags που χρησιμοποιουνται για την παρατηρηση της καταστασης του παιχνιδιου οπως το in_use και το character mode παιρνουν τις τιμες που τους αναλογουν.Αν ο χαρακτηρας εχει ολοκληρωσει τις ενεργειες του τοτε το character mode γινεται false και το in_use που αποθηκευει τον εν χρηση χαρακτηρα χρησιμοποιειται για να ανατεθει η τιμη true στο πεδιο used του εκαστοτε χαρακτηρα.

- Η συναρτηση defaul χρησιμοποιει το action command του κουμπιου που πατηθηκε για να βρει το tile που του αντιστοιχει απο το hash map .Υστερα αφου γινουν οι καταλληλοι ελεγχοι για το ποσα πλακιδια εχει παρει ο παικτης(δεν επιτρεπεται πανω απο δυο),και της περιοχης ππου εχει διαλεξει(πρεπει ολα να ειναι απο την ιδια) ενημερωνεται το πεδιο του board που αντιστοιχει στην περιοχη απο την οποια ο παιχτης πηρε το πλακιδιο και το πλακιδιο μπαινει στα κατεχομενα πλακιδια του παιχτη.Στη συνεχεια βρισκουμε το card στο player container που ειναι visible ωστε να τοποθετησουμε πανω το πλακιδιο που πηρε ο παιχτης .Το πεδιο tiles taken του παιχτη αυξανεται κατα 1 σε καθε τετοια ενεργεια ωστε να ξερουμε ποσα πλακιδια εχει παρει(αν προσπαθησει να παρει πανω απο 2 εμφανιζεται μηνυμα λαθους),επιπλεον στο πεδιο chosen area ανατιθεται η τιμη της περιοχης απο την οποια παρθηκαν τα πλακιδια.

- `private class Menu_actions implements ActionListener`//η κλάση υλοποιεί την λειτουργια του menu ,οι επιλογες save και load δεν ειναι διαθεσιμες ,σε περιπτωση που πατηθει το exit game εμφανιζεται παραθυρο το οποιο ζηταει για επιβεβαιωση της επιλογης,σε περιπτωση που γινει η επιβεβαιωση η εφαρμογη κλεινει.

- `private class Player_Actions implements ActionListener`//η κλαση υλοποιει τις

λειτουργιες των κουμπιων draw και end turn αντιστοιχα.

- Αν ο παικτης πατησει το κουμπι draw τοτε τοπθετουνται 4 πλακιδια απο την τσαντα στο ταμπλο,στη συνεχεια διαγραφονται απο την τσαντα.Υστερα γινεται ελεγχος για το αν η εισοδος εχει 16 πλακιδια κατολισθησης (αν ισχυει αυτο το παιχνιδι τελειωνει).Αν ο παικτης τραβηξει πλακιδιο κατολισθησης και το παιχνιδι ειναι σε single player mode καλειται ο grave robber.Τελος η τσαντα γινεται shuffle και το πεδιο has drawn του παιχτη γινεται true ωστε να αποφευχθει το να τραβηξει ξανα.

- Αν ο παιχτης πατησει το κουμπι end turn τοτε τα flags τα οποια χρησιμοποιουνται για ελεγχο των ενεργειων παιρνουν default values ,για παραδειγμα τα tiles taken για ολους τους παικτες γινονται 0 το character_used_for_turn γινεται false(ελεγχει αν ο παικτης εχει χρησιμοποιησει χαρακτηρα,επιτρεπεται ενας ανα γυρο) ,ο γυρος παιρνει το index που το αναλογει και το πεδιο has drawn του παικτη του οποιου ειναι η σειρα γινεται false,επισης εμφανιζεται το card που αναλογει στον παικτη(τα cards εχουν τα ονοματα των παικτων για να ειναι ευκολη η επιλογη τους).

Η ΚΛΑΣΗ Main_menu:

Η κλάση Main_menu υλοποιεί την γραφική διεπαφή που συναντά ο χρήστης μόλις εκτελέσει το παιχνίδι.Διαθέτει τις επιλογές New Game,Load Game και Exit Game.Η κλάση διαθέτει τα παρακάτω χαρακτηριστικά.

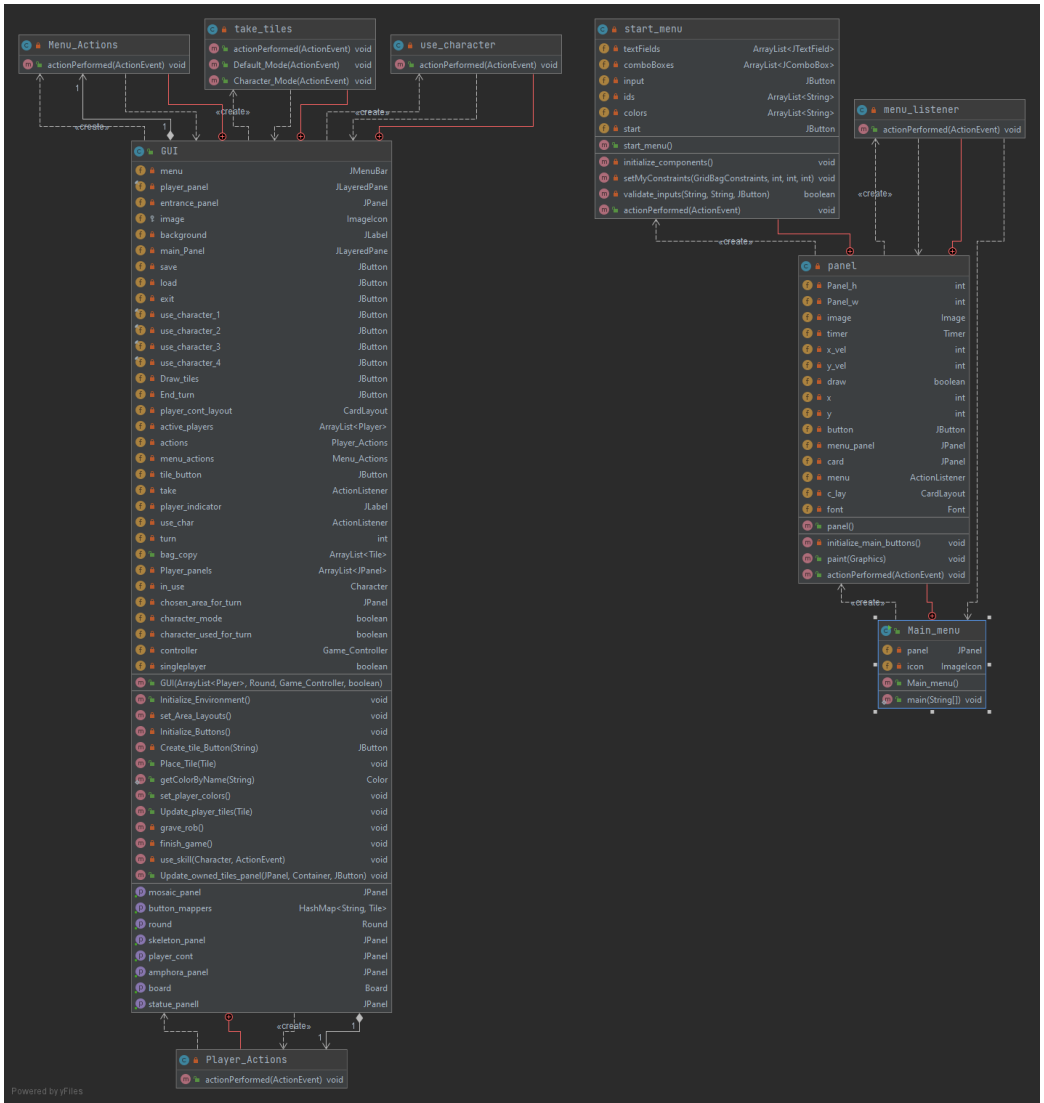
- `public Main_menu()//`Κατασκευαστρια μεθοδος η οποια δημιουργει το main

frame ,θετει τις διαστασεις του και αρχικοποιει τα υπολοιπα components

- Η κλάση περιέχει τις παρακάτω εμφολευμένες κλάσεις
- private class panel extends JPanel implements ActionListener η οποία υλοποιεί το Jpanel που τοποθετείται στο frame και έχει τις εξής μεθόδους:
- public panel()//Κατασκευαστρια μεθοδος που αρχικοποιει τα πεδια της κλασης ,δημιουργει το panel με διαστασεις ,αρχικοποιει paths για εικονες και για το font που θα χρησιμοποιηθει στα κουμπια οπως και τις διαστασεις και την θεση του καθε panel(menu_panel,card).Το panel εχει card layout ωστε με το πατημα καθε κουμπιου να μεταβαινει ο χρηστης σε διαφορετικο card.Σημειωση:Το main panel ειναι διαφανο,υπαρχει το menu panel το οποιο εχει τα κουμπια new game,load game,exit και το layout ειναι boxlayout.
- private void initialize_main_buttons()//Κατασκευαζει τα κουμπια που θα τοποθετηθουν στο main panel,τους δινει διαστασεις,εικονες,font ,action listener(menu) και action commands.
- public void paint(Graphics g)//εμφανιζει στο panel την κινουμενη εικονα.
- public void actionPerformed(ActionEvent e)//Επαναυπολογιζει τις συντεταγμενς στις οποιες θα ζωγραφιστει η εικονα πανω στο panel κανοντας χρηση του timer και της paint.

- Η κλάση `private class start_menu extends JPanel implements ActionListener` //Ειναι εμφολευμενη στην κλαση Panel και ειναι υπευθυνη για την δημιουργια και λειτουργικοτητα του menu που δεχεται Inputs των αναγνωριστικων του παικτη και βασει αυτων ξεκινανε το παιχνίδι.Για το layout εχει χρησιμοποιηθει το gridbag layout ωστε να τοποθετηθουν καταλληλα τα buttons,comboboxes και textboxes,Τελος γινονται οι καταλληλοι ελεγχοι εγκυροτητας των inputs ωστε να μην μπορουν να υπαρχουν δυο οι παραπανω παικτες με το ιδιο id και χρωμα κτλ.

UML ΔΙΆΓΡΑΜΜΑ ΤΟΥ ΠΑΚΈΤΟΥ VIEW



- UML ΔΙΆΓΡΑΜΜΑ ΤΟΥ PROJECT