ΣΧΕΔΙΑΣΜΟΣ PROJECT

Το project θα βασιστεί πάνω στο μοντέλο MVC. Άρα ο σκοπός είναι ο controller να αποτελεί τον συνδετικό κρίκο με το Model και το View. Θα δείξουμε τα πακέτα του Model και του Controller καθώς και μια αναφορά στα περιεχόμενα του View.

PACKAGE MODEL

Σε αυτό το πακέτο περιέχονται η abstract κλάση Card, οι κλάσεις που την κληρονομούν TrainCard και PointsCard με τις υποκλάσεις τους που θα αναλυθούν παρακάτω. Περιέχονται ακόμα οι κλάσεις που είναι βασικές για την λειτουργία του παιχνιδιού, όπως η Player, η Railyard και η OnTheTrack. Αναλυτικότερα έχουμε:

Card Classes:

Public Abstract Class Card: Αυτή η αφηρημένη κλάση είναι υπεύθυνη για τις κάρτες του παιχνιδιού και αποτελείται απο τις βασικές πληροφορίες που πρέπει να περιέχει κάθε κάρτα.

Τα attributes:

1. **private Boolean is_active:** αποθηκεύει αν μια κάρτα έχει χρησιμοποιηθεί η όχι
2. **String belongs:** αποθηκεύει σε ποιον παίκτη ανήκει

Οι μέθοδοι:

1. **public void change_active(Card given):** Αλλάζει το αν μια κάρτα έχει χρησιμοποιηθεί η όχι
2. **public void card_owner(Card given, String name):** Αλλάζει σε τον ιδιοκτήτη της κάρτας.
3. **public Boolean is_active(Card given):** Επιστρέφει το αν μια κάρτα έχει χρησιμοποιηθεί
4. **public String is_owner(Card given):** Επιστρέφει τον ιδιοκτήτη μιας κάρτας
5. **public Card(String belong, Boolean active):** Ο πρώτος εκ των 2 constructor, κατά του οποίου το κάλεσα έχουμε και αρχικοποίηση attributtes
6. **public Card():** Ο default constructor της κλάσης

Public Class PointsCard extends Card: Αυτή είναι η κλάση που αντιπροσωπεύει τις κάρτες πόντων, είναι υποκλάση της Card.

Τα attributes:

1. private int points: αποθηκεύει τους πόντους του κάθε παίκτη

2. String start_city: αποθηκεύει το όνομα της πρώτης πόλης που αναγράφεται πάνω στην κάρτα

Οι μέθοδοι:

1. public void set_start_city(PointsCard given, String cit): Αλλάζει το όνομα της αρχικής πόλης πάνω στην κάρτα

2. public String get_start_city(PointsCard given): Επιστρέφει το όνομα της αρχικής πόλης της κάρτας

3. public void set_points(PointsCard given, int po): Αλλάζει τους πόντους που αναγράφονται πάνω στην κάρτα

4. public int get_points(PointsCard given): Επιστρέφει τους πόντους που αναγράφονται πάνω στην κάρτα

5. public PointsCard(String owner, Boolean active, int point, String name): Ο πρώτος εκ των 2 constructor, κατά του οποίου το κάλεσα έχουμε και αρχικοποίηση attributtes

6. public PointsCard(): Ο default constructor της κλάσης

Public Class TrainCard extends Card: Αυτή είναι η κλάση που αντιπροσωπεύει τις κάρτες τραίνων, είναι υποκλάση της Card.

Τα attributes:

1. private String color: αποθηκεύει το χρώμα της κάρτας

Οι μέθοδοι:

1. public Boolean check_balader(Card given): Ελένχει αν η κάρτα είναι μπαλαντέρ

2.public void set_color(TrainCard given, String col): Αλλάζει

Το χρώμα της κάρτας στο δοσμένο χρώμα

3. public String get_color(Card given): Επιστρέφει το χρώμα της κάρτας

4. public TrainCard(String color, String owner, Boolean active): Ο πρώτος εκ των 2 constructor, κατά του οποίου το κάλεσα έχουμε και αρχικοποίηση attributtes

5. public Card(): Ο default constructor της κλάσης

Public Class BigCitiesCard extends Card: Αυτή είναι η κλάση που αντιπροσωπεύει τις κάρτες Big Cities, είναι υποκλάση της PointsCard.

Τα attributes:

1. private int visit_count: αποθηκεύει τον αριθμό που έχει επισκεφθεί ο αντίστοιχος παίκτης την πόλη της κάρτας

Οι μέθοδοι:

1. public void set_visit(BigCitiesCard given, int numb): Αλλάζει τον αριθμό των επισκέψεων στον δοσμένο

2. public void int get_visit(BigCitiesCard given): Επιστρέφει την visit_count

3.public BigCitiesCard(String color, String owner, boolean active,int count, int points, String name): Ο πρώτος εκ των 2 constructor, κατά του οποίου το κάλεσα έχουμε και αρχικοποίηση attributtes

4. public BigCitiesCard(): Ο default constructor της κλάσης

Public Class DestinationCard extends Card: Αυτή είναι η κλάση που αντιπροσωπεύει τις κάρτες προορισμού, είναι υποκλάση της PointCard.

Τα attributes:

1. private String end_city: αποθηκεύει την τελική πόλη προορισμού

Οι μέθοδοι:

1. public void set_end_city(DestinationCard given, String cit): Αλλάζει την τελική πόλη προορισμού ανάλογα με την δοσμένη
2. public String get_end_city(DestinationCard given): Επιστρέφει την τελική πόλη προορισμού
3. public DestinationCard(String color, String owner, boolean active,int count, int points, String first_cit, String sec_cit):Ο πρώτος εκ των 2 constructor, κατά του οποίου το κάλεσα έχουμε και αρχικοποίηση attributtes
4. public DestinationCard():Ο default constructor της κλάσης

public Class OnTheTrack: Αυτή είναι η κλάση που υλοποιεί τις λειτουργίες οι οποίες σχετίζονται με το ταμπλό

Δεν υπάρχουν attributes

Οι μέθοδοι:

1. public void collectFromRailYard(RailYard target): η μέθοδος που υλοποιεί το πέρασμα καρτών στο ταμπλό

2. public void buyDestinationCard(Card given): Η μέθοδος που υλοποιεί την αγορά καρτών

3. public OnTheTrack(): o default constructor της κλάσης

public Class Player: Η κλάση που υλοποιεί τον παίκτη

Τα attributes:

1. **private ArrayList<Card> hand: Αποθηκεύει τις κάρτες που έχει ο παίκτης στο χέρι του**
2. **private TrainCard my_train_cards[] = Αποθηκεύει τις κάρτες τραίνων του παίκτη**
3. **private DestinationCard my_destination_cards[]: Αποθηκεύει τις κάρτες προορισμού του παίκτη**
4. **private BigCitiesCard my_bigcities_cards[]: Αποθηκεύει τις κάρτες BigCities του παίκτη**
5. **private RailYard my_yard: Αποθηκεύει το RailYard του παίκτη**
6. **public OnTheTrack my_track: Αποθηκεύει την κατάσταση του track**
7. **private int score: Αποθηκεύει το σκόρ του παίκτη**

Οι μέθοδοι:

1. **public int get_score(): επιστρέφει το σκόρ του παίκτη**

2.public void set_score(int score): Αλλάζει το σκόρ του παίκτη με το δοσμένο

3. public RailYard get_My_yard(): Επιστρέφει το RailYard του παίκτη

4. public void setMy_yard(RailYard my_yard): Αλλάζει το RailYard του παίκτη

5. public BigCitiesCard[] getMy_bigcities_cards(): Επιστρέφει την αρχή των BigCity Cards του παίκτη

6. public DestinationCard[] getMy_destination_cards(): Επιστρέφει την αρχή των Destination Cards του παίκτη

7. public TrainCard[] getMy_train_cards(): Επιστρέφει την αρχη των Train Cards του παίκτη

8. public OnTheTrack getMy_track(): Επιστρέφει την τρέχοντα κατάσταση του track

9.public void setMy_track(OnTheTrack my_track): Αλλάζει το track του χρήση με το δοσμένο

10.public ArrayList<Card> getHand(): Επιστρέφει το χέρι με τις κάρτες του χρήστη

public class RailYard: Η κλάση η οποιά υλοποιεί το RailYard κάθε παίκτη με τις λειτουργίες του

Τα attributes:

1. private List<Card>[] cards: οι λίστες των καρτών που υπάρχουν στο RailYard του παίκτη

Οι μέθοδοι:

1. public void add_card(Card given): Η μέθοδος η οποία υλοποιεί τη μεταφορά καρτών στο RailYard

2. public Card get_card(Card given): Η μέθοδος η οποία υλοποιεί τη μεταφορά καρτών εκτός του RailYard

3. public void playCards(List<Card> list, int index, RailYard enemy): Η μέθοδος η οποία υλοποιεί την ρίψη καρτών απο τον παίκτη

4. public void Train_Rob(List<Card> list, int index, RailYard enemy): Η μέθοδος η οποία υλοποιεί περίπτωση train robbing

5. public List<Card>[] getCards(): Μέθοδος η οποία επιστρέφει την αρχή σε κάποια απο τις λίστες καρτών

6. public RailYard(): ο default constructor

public class Deck: η κλάση η οποία υλοποιεί το συνολο των καρτών

Τα attributes:

1. public List <Card> deck: Υλοποιεί το σύνολο των καρτών στο deck

2. public TrainCard[] middle: Αποθηκεύει τις κάρτες στη μέση του track

3. public List<BigCitiesCard> big: Αποθηκεύει τις κάρτες Bigcities του deck

Οι μέθοδοι:

public Deck(): default constructor

Ακολουθεί ανάλυση του controller και των στοιχείων που τον αποτελούν

public Class Controller: η βασική κλάση η οποία συνδέει το Model με το View

Τα attributes:

1. public Deck all-cards: η σύνδεση του controller με το deck
2. private Player P1,P2 = οι δύο παίκτες

Οι μέθοδοι:

1. public void setCards(): Η μέθοδος η οποία υλοποιεί τους 3 τύπους καρτών για κάθε παίκτη
2. public void set_deck(): Η μέθοδος η οποία αρχικοποιεί το deck

3. public void start(): Η μέθοδος η οποία επιλέγει παίκτη και αρχίζει το παιχνίδι

4. public void distribute(): Η μέθοδος η οποία μοιράζει τις κάρτες στους παίκτες

5. public void hand(): Η μέθοδος η οποία σετάρει τις κάρτες του χεριού του παίκτη

6. public void removecard(Card given): η μέθοδος η οποία διαγράφη κάρτες είτε απο το deck η\και απο το χέρι του παίκτη

7. public void playnormalCard(Card given): η μέθοδος η οποία υλοποιεί τη ρήψη μιας κάρτας

8. public void check_valid_card(Card given): Η μέθοδος η οποία τσεκάρει αν μπορεί κάποια\οιες κάρτες να παιχτούν

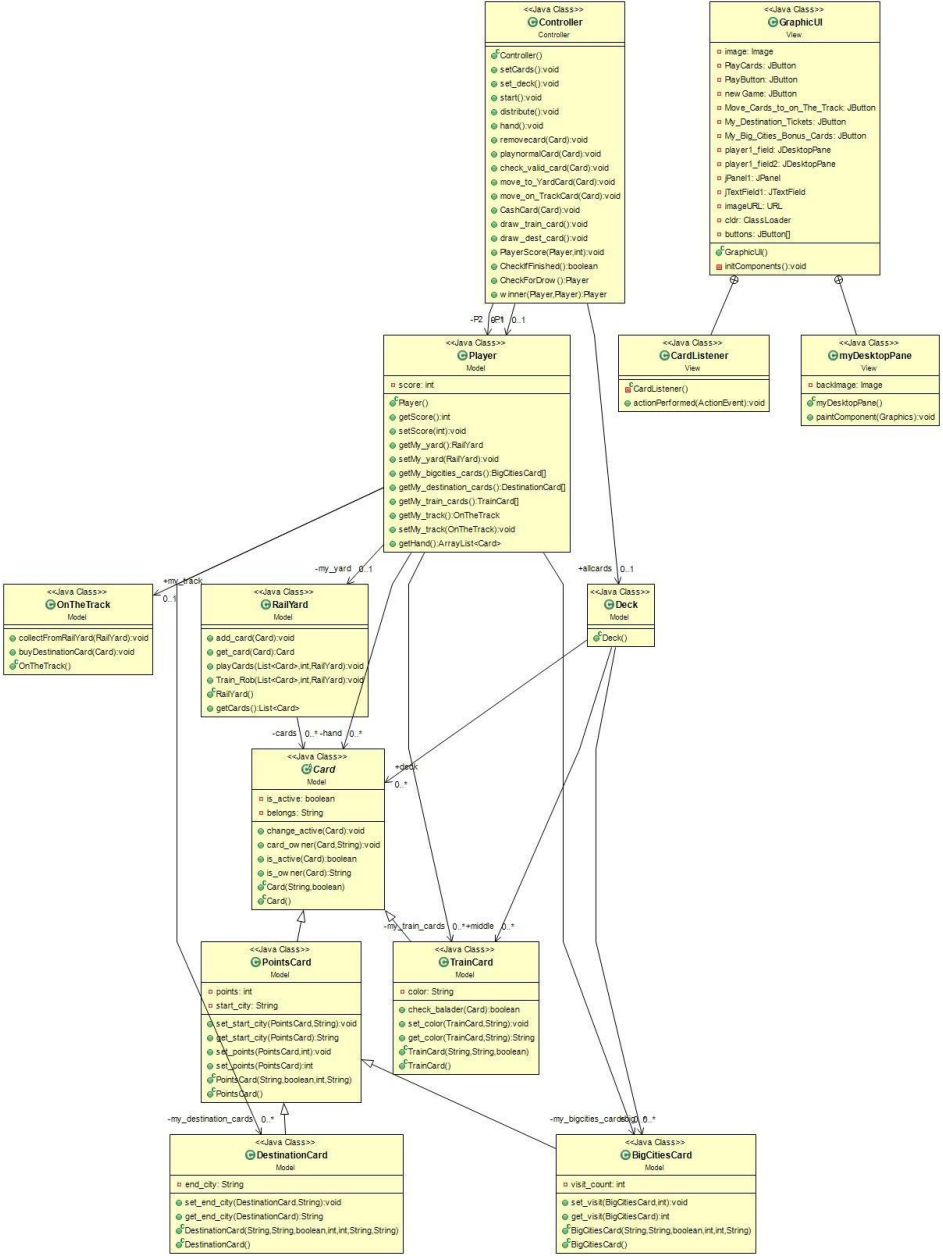
9. public void move_to_YardCard(Card given): η μέθοδος η οποία μεταβιβάζει μια κάρτα στο RailYard κάποιου παίκτη

10. public void move_on_TrackCard(Card given): η μέθοδος η οποία μεταβιβάζει μια κάρτα στο track

11. public void CashCard(Card given): Η μέθοδος η οποία υλοποιεί την εξαργύρωση μιας κάρτας

12. public void draw_train_card(): Η μέθοδος η οποία υλοποιεί το τράβηγμα μιας κάρτας train απο το deck

13. public void draw_dest_card(): Η μέθοδος η οποία υλοποιεί το τράβηγμα μιας κάρτας προορισμού απο το deck\



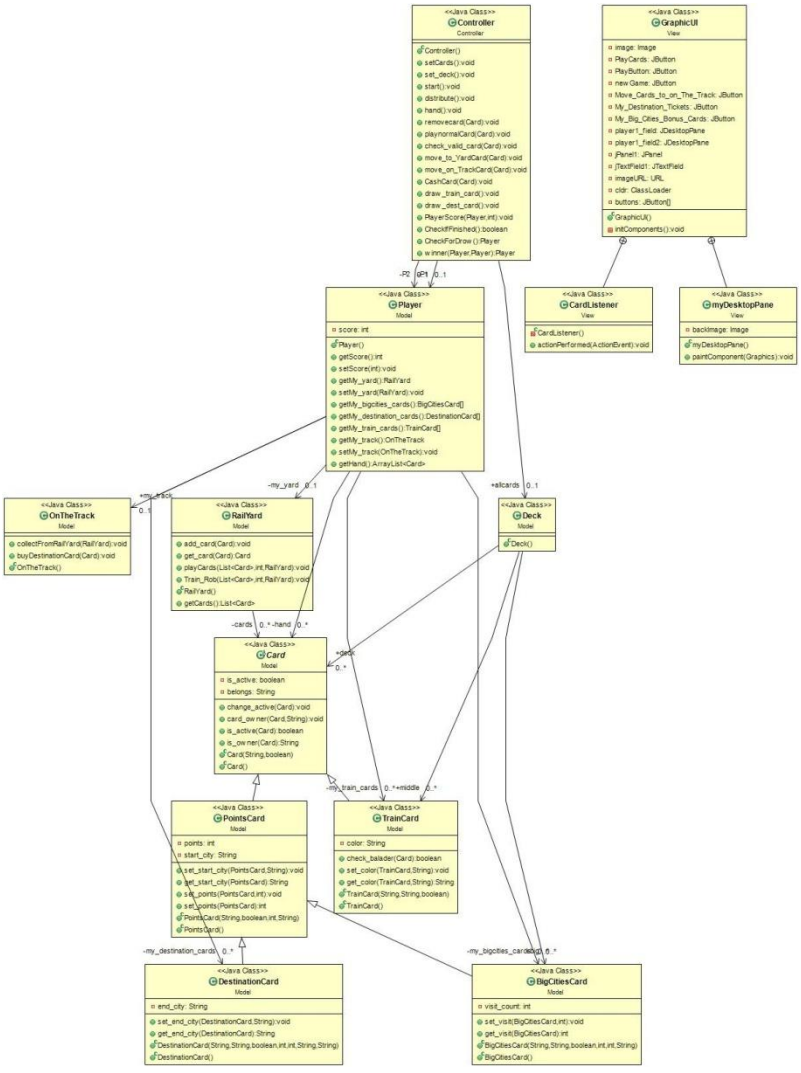
14. **public void PlayerScore(Player p, int score): Η μέθοδος η οποία υλοποιεί την αναβάθμιση του σκορ ενός παίκτη**

15. **public boolean CheckIfFinished(): Η μέθοδος η οποία τσεκάρει για το αν το παιχνίδι έχει τελιώσει**

16. **public Player CheckForDrow(): Η μέθοδος η οποία ελένχει αν το παιχνίδι είναι ισοπαλία**

17. **public Player winner(Player p1, Player p2): Η μέθοδος η οποία ελένχει και επιστρέφει τον νικητή του παιχνιδιού**

Ακολουθεί διάγραμμα UML το οποίο δείχνει τις συνδέσεις μεταξύ Model, View, Controller



Αναφορικά. Η View θα περιέχει το γραφικό περιβάλλον το οποίο θα ελένχεται απο τον Controller. Υλοποιόντας τα κουμπία το ταμπλό και τα pop-up μηνύματα