OBJECT-ORIENTED PROGRAMMING HY-252

PROJECT 2012-2013

Michalis Moundandonakis Dimitrios Xanthakis

Σχεδιασμός της Εργασίας

Η υλοποίηση της εργασίας θα βασιστεί πάνω στο μοντέλο MVC (Model View Controller).Έτσι , σκοπός μας είναι ο Controller να είναι ο συνδετικός κρίκος των Model και view. Οπότε στη συνέχεια της αναφοράς μας θα αναλύσουμε λίγο ιδιαίτερα τα κομμάτια του Model και Controller που είναι σημαντικά για αυτή τη φάση και τέλος θα αναφερθούμε και λίγο στο view.

Package Model

Σε αυτό το πακέτο θα περιέχονται διεπαφή Card,οι κλάσεις CardColor, SimpleCard και SpecialCard, κλάσεις που κληρονομούν την Special Card , οι κλάσεις Team, Player και Sullogi, η κλάση Turn και τέλος οι κλάση Round και οι τύποι γύρων (κλάσεις που κληρονομούν την Round).

Card Interface and Other Classes for Cards

Αρχικά φτιάχνοντας τη διεπαφή Card μας δίνεται η δυνατότητα να προσπελάσουμε τα δεδομένα χωρίς να πρέπει να ορίσουμε αν μία κάρτα είναι απλή η ειδική.

Το interface αυτό μας παρέχει τις εξής μεθόδους:

1. public int getValue(); Accessor (Selector)

   Returns the value of a Card.

2. public void setValue(int value);Transformer (Mutative)

   Sets the value of a card.

3. public int getPoints(); Accessor (Selector)

   Returns the points of a card.

4. public void setPoints(int points); Transformer (Mutative)

   Sets the points of a card.

5. public String toString();

   Returns a String Representation of a Card.

6. public void SetTempValue(int j); Transformer (Mutative)

   Sets the temp value of a card.

7. public int GetTempValue();Accessor (Selector)

   Returns the temp value of a card.

8. public CardColor getColor();Accessor (Selector)

   Returns the card's color

Στη συνέχεια έχουμε την SimpleCard και την SpecialCard που υλοποιούν την Card.

Class SimpleCard

Εδώ θα αναφέρουμε τα attributes και τις υπόλοιπες μεθόδους που έχει η κλάση αυτή (εκτός από αυτές που υλοποιεί μέσω της διεπαφής Card).

Τα attributes:

1) 1.private CardColor col; //The color of a Simple Card.
2) 2.private int value; //The value of a Simple Card.
3) 3.private int points; //The points of a Simple Card.

Οι υπόλοιπες μέθοδοι:

1. public CardColor getColor(); Accessor (Selector)

   Get the color of a simple Card.

2. public void setColor(CardColor col); Transformer (Mutative)

   Sets the color of a simple Card.

Class CardColor

Είναι μια enum class που περιέχει τα χρώματα των καρτών(BLUE,GREEN,BLACK,RED)

Class SpecialCard

Εδώ θα αναφέρουμε τα attributes και τις υπόλοιπες μεθόδους που έχει η κλάση αυτή (εκτός από αυτές που υλοποιεί μέσω της διεπαφής Card).

Τα attributes:

1) private int temp_val; //The temp value of a Special Card.
2) private int value; //The value of a Special Card.
3) private int points; //The points of a Special Card.
4) private String type; //The type of a Special Card

Οι υπόλοιπες μέθοδοι:

1. public void setType(String type) ; Transformer (Mutative)

   Sets the type of the Special Card.

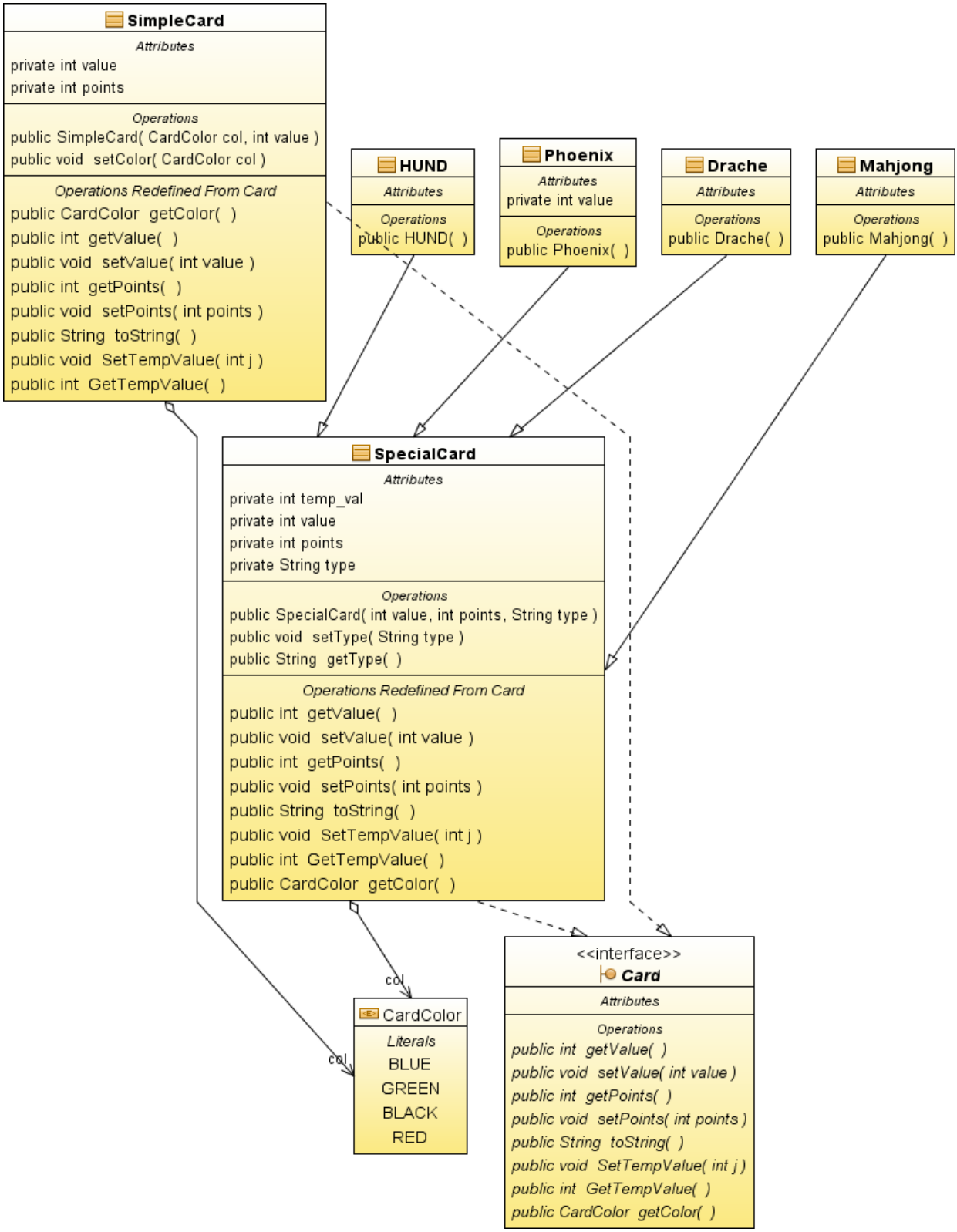
2. public String getType() ;

   Return the type of the Special Card.

Classes Mahjong Phoenix Hund Drache

Αυτές οι κλάσεις κάνουν extend την Special Card και μέσω της εντολής super αποκτούν πρόσβαση στη κλάση Special Card και αρχικοποιούν τις τιμές value type και points.

- Τέλος, εδώ θα δείξουμε μια αναπαράσταση των κλάσεων που έχουν σχέση με τις κάρτες μέσω UML.


SimpleCard
Attributes
private int value
private int points
Operations
public SimpleCard(CardColor col, int value)
public void setColor(CardColor col)
Operations Redefined From Card
public CardColor getColor( )
public int getValue( )
public void setValue(int value)
public int getPoints( )
public void setPoints(int points)
public String toString( )
public void SetTempValue(int j)
public int GetTempValue( )
HUND
Attributes
Operations
public HUND( )
Phoenix
Attributes
private int value
Operations
public Phoenix( )
Drache
Attributes
Operations
public Drache( )
Mahjong
Attributes
Operations
public Mahjong( )
SpecialCard
Attributes
private int temp_val
private int value
private int points
private String type
Operations
public SpecialCard(int value, int points, String type)
public void setType(String type)
public String getType( )
Operations Redefined From Card
public int getValue( )
public void setValue(int value)
public int getPoints( )
public void setPoints(int points)
public String toString( )
public void SetTempValue(int j)
public int GetTempValue( )
public CardColor getColor( )
col
col
CardColor
Literals
BLUE
GREEN
BLACK
RED
<<interface>>
Card
Attributes
Operations
public int getValue( )
public void setValue(int value)
public int getPoints( )
public void setPoints(int points)
public String toString( )
public void SetTempValue(int j)
public int GetTempValue( )
public CardColor getColor( )

Class Round

Η κλάση αυτή θα είναι υπεύθυνη για το να τσεκάρει αν μία συλλογή μπορεί να παιχτεί ανάλογα με τον τύπο γύρο που έχουμε. Συγκεκριμένα θα εξετάζει αν μία συλλογή είναι έγκυρη σύμφωνα με τους κανόνες του παιχνιδιού και στη συνέχεια μέσω των υποκλάσεων που την κληρονομούν θα γίνεται η σύγκριση με την προηγούμενη συλλογή για το αν τελικά μπορεί να παιχτεί ή όχι η συλλογή.

Τα attributes της κλάσης Round:

1) private int type; //type of the round (for example pairs,fullhouse…)
2) private boolean bomb_round; // if a round is a bomb round
3) private boolean roundStarted; // if a round has started
4) private Boolean selection; //if player made a selection for phoenix value
5) private Sullogi cards = new Sullogi(); // a collection that a player wants to play
6) private Sullogi roundcards=new Sullogi(); // all the cards that have been played in the round
7) private int euxi; //wish

Οι μέθοδοι:

1. public void setCardsToPlay(Sullogi cards);Transformer(Mutative)

   Sets the Cards that a player wants to play

2. public void setroundtype();Transformer(Mutative)

   Sets the type of the round

3. public int getRoundType();Accessor(Selector)

   Returns the type of the round;

4. public boolean checkFull(); Observer

   Check if a Sullogi is Full House and if it is Full House return true

5. public boolean checkStraight(); Observer

   Check if a Sullogi is Straight and if it is Straight return true

6. public boolean compareCollections(Sullogi other); Observer

   Check if a Sullogi can be played (if is better than the previous that have been played) and if can be played return true

7. public boolean checkPairs(); Observer

   Check if a Sullogi is Pairs and if it is Pairs return true

8. public boolean checkBomb(); Observer

   Check if a Sullogi is Bomb and if it is Bomb return true.

9. public boolean bomb_round(); Observer

   Returns true if it is a bomb Round false otherwise.

10. public void Set_bomb_round(); Transformer (Mutative)

    Sets the bomb round to true.

11. public void roundStarted(); Transformer (Mutative)

    Starts the round

12. public boolean getRoundStarted(); Observer

    Returns true if a Round has started, false otherwise.

13. public void Update_Cards(Sullogi other); Transformer (Mutative)

    Update the roundcards when a player has just played

14. public Sullogi GetRoundCards(); Accessor(Selector)

    Returns the collection of all the cards in the round

15. public void SetSelection(); Transformer (Mutative)

    If a player have select a value for phoenix,set selection variable to true

16. public boolean SelectPhoenixValue(); Observer

    Returns true if a player has made a selection for phoenix value, otherwise false

17. public void SetDracheInRound(); Transformer

    If this round has drache set this.drache to true

18. public boolean DracheInRound();Observer

Returns true if there is drache in round

19. public void lastCards(Sullogi lastCards); Transformer

    Saves the last Cards that have been played

20. public String [] GetlastCards(); Accessor

    Returns the last cards that have been played as a string array

21. public void SetEuxi(int k); Transformer

    Sets euxi to k

22. public int GetEuxi() ; Accessor

    Returns the wish(euxi)

23. public boolean CheckEuxi(); Observer.

    Returns true if there is a card with wish value in cards

24. public boolean BombHasEuxi(Sullogi a); Observer

    Returns true if there is a bomb with euxi value in cards

25. public void SetPlayerAllCards(Sullogi a); Transformer

    Sets players all cards to Sullogi Playerallcards

26. public Sullogi GetPlayerAllCards(); Accessor

    Returns all the cards of a player

27. public boolean EuxiMeBomb(); Observer

    Returns true if there is a bomb with euxi value in player all cards

Classes which extends Round

- Bomb
- FullHouse
- Monofyllia
- Pairs
- SimplePair
- Straight
- Tripleta

Κάθε κλάση έχει δύο attributes:

- Private Sullogi lastCardsPlayed=new Sullogi(); //the last Cards collection that have been played
- boolean haswish; // if a player has a wish in the cards which wants to play

Εκτός από τη Monofyllia που έχει:

- Card lastCardPlayed=new Card(); //the last card that has been played
- boolean haswish; // if a player has a wish in the cards which wants to play

Επίσης οι Straight , Pairs και Bomb έχουν ακόμα ένα attribute:

- Private int size //the number of the cards in the round

Τέλος κάθε κλάση κάνει override τρεις μεθόδους από την parent class:

- public boolean compareCollections(Sullogi other);
  Transformer(Mutative)

  Check if a Sullogi can be played (if is better than the previous that have been played) and if can be played saves it as the lastCardsPlayed Sullogi and return true

- public void SetIfWishChanged();Transformer(Mutative)

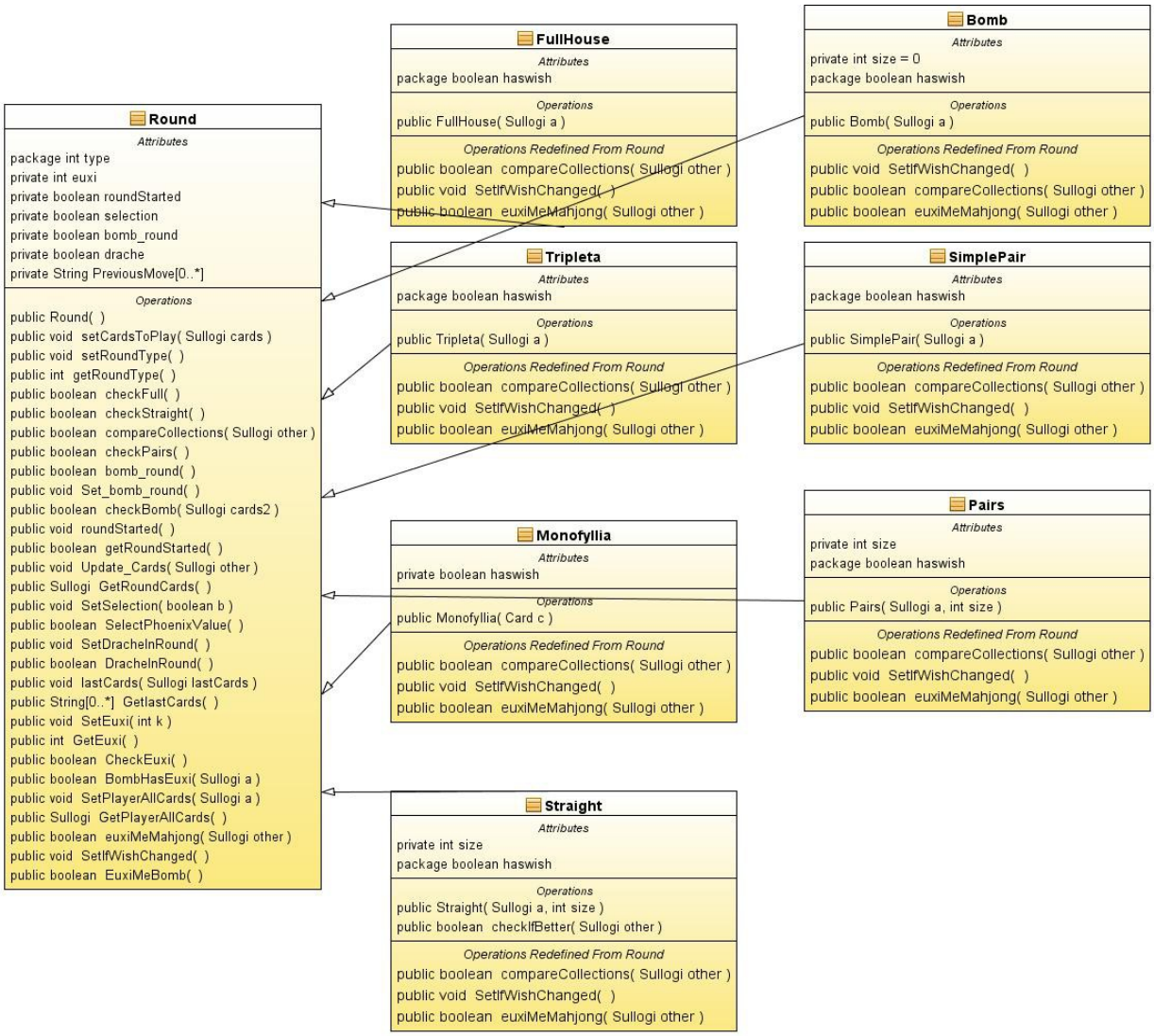
  Check if a wish has played and set euxi value to 0

- public boolean euxiMeMahjong(Sullogi other); Transformer(Mutative)

  Checks if player has a better Sullogi which includes wish in his cards

✓ Σημείωση: το override της μεθόδου μας δίνει τη δυνατότητα να έχουμε ένα αντικείμενο τύπου Round και να κάνουμε broadcast σε αντικείμενα υποκλάσεων της.

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



Class Sullogi

Η κλάση αυτή ουσιαστικά διαχειρίζεται μια Arraylist και μας δίνει τη δυνατότητα να φτιάξουμε συλλογές από κάρτες. Επίσης μέσα σε αυτή την κλάση υπάρχουν συναρτήσεις που αρχικοποιούν τις 56 κάρτες και κάνουν sort σε οποιαδήποτε arraylist για όσες θέσεις επιθυμούμε.

Έχει ένα attribute:

- private ArrayList<Card> cards = new ArrayList<Card>() // a Card ArrayList

Και έχει τις εξής μεθόδους:

1. public void init_cards(); Transformer(Mutative)

   Initialize the 56 cards

2. public void sort(int i, int j); Transformer(Mutative)

   It is a sort void, which sort an arraylist from position i to j-1

3. public boolean isEmpty(); Observer

   Returns if an ArrayList is empty

4. public void addCard(Card i); Transformer(Mutative)

   Add a Card to ArrayList

5. public int getValue(int i); Accessor (Selector)

   Returns the value of a Card in position i

6. public void removeCard(Card i); Transformer(Mutative)

   Removes a Card from ArrayList

7. public int size();Accessor (Selector)

   Returns the size of the ArrayList

8. public Card getCard(int index) ; Accessor (Selector)

   Returns a Card in position index in the ArrayList

9. public void clearAll();Transformer(Mutative) ; Accessor (Selector)

   Remove all the elements from the ArrayList

10. public ArrayList<Card> getCards();Accessor (Selector)

    Returns a Card ArrayList

Class Turn

Η κλάση αυτή ουσιαστικά διαχειρίζεται τη σειρά των παιχτών μέσα στο παιχνίδι.

Τα attributes:

1) private int currentID; // the ID of the player who has now the turn
2) private int num; // the number of the players
3) private int last_player; // the ID of the last player who dropped cards
4) Private int round_players; // the round players

- ✓ Σημείωση: η μεταβλητή round_players και κάποιες μέθοδοι που την χρησιμοποιούν παρακάτω χρειάζονται γιατί μπορεί ένας παίχτης να έχει βγει αλλά κανένας άλλος παίκτης να μην έχει ρίξει καλύτερο συνδυασμό από αυτόν οπότε ουσιαστικά θα πάρει αυτός τη μπάζα. Οπότε ουσιαστικά αν δε ρίξει κάποιος άλλος καλύτερο συνδυασμό το φύλλο του ακόμα παίζει παρ' όλο που έχει βγει

Οι μέθοδοι:

1. public void setID(ArrayList <Player> players);Transformer

   Sets which player has the turn to play

2. public int getID(); Accessor(Selector)

   Returns the player's ID whose turn is to play

3. public boolean checkIfPlayerFinished(Player p); Observer

Returns true if a player has finished, else false

4. public void NumberOfPlayers(); Transformer(Mutative)

   Sets the number of players.

5. public int GetNumberOfPlayers(); Accessor(Selector)

   Returns the number of players

6. public int Get_Round_Players(); Accessor(Selector)

   Returns the number of players in this round

7. public void SetRoundPlayers(int j) ; Transformer(Mutative)

   Sets the number of round players.

8. public void Set_last_player(int k) ; Transformer(Mutative)

   Sets the most recent player, who has dropped cards in the table.

9. public int Get_last_player(); Accessor(Selector)

   Returns the most recent player, who has dropped cards in the table

Class Team

Η κλάση αυτή αναπαριστά μια ομάδα μέσα στο παιχνίδι.

Τα attributes:

1) private String name; // the name of the team
2) private int points; // the points of the team

Οι μέθοδοι

1. public String getName(); Accessor (Selector)

   Returns the name of the team

2. public void setName(String newName); Transformer(Mutative)

   Sets the name of the team

3. public int getPoints(); Accessor(Selector);

   Returns the points of the team

4. public void setPoints(int points); Transformer(Mutative)

   Sets the points of the team

Class Player

Η κλάση αυτή αναπαριστά ένα παίκτη μέσα στο παιχνίδι.

Τα attributes:

1) private String name; // the name of the player
2) private Team team; //the team of the player
3) private Sullogi cards, cards_to_play, mpaza; // collections of player
4) private int choice,ID; // choice of playes(tichu , GrandTichu) and player's ID
5) private boolean hasPlayed,finished; //hasPlayed is a variable who changes if a player has dropped cards at least one time and finished if player's Cards Sullogi is empty.

Οι μέθοδοι:

1. public void init_player(); Transformer(Mutative)

   Initializes a player for a new deal(moirasma)

2. public int getID(); Accessor (Selector)

   Returns the ID of a player

3. public void setID(int id); Transformer(Mutative)

   Sets the ID of a player

4. public String getName(); Accessor (Selector)

   Returns the name of the player

5. public void setName(String newName); Transformer(Mutative)

   Sets the name of the player

6. public Team getTeam(); Accessor (Selector)

   Returns the team of the player

7. public void setTeam(Team team); Transformer(Mutative)

Sets the team of the player

8. public void setCards_to_play(Sullogi cardsforplaying); Transformer

   Sets the collection(Sullogi) who wants a player to play

9. public Sullogi getCards_to_play(); Accessor (Selector)

   Returns the cards who wants a player to play

10. public void setCards(Card c); Transformer(Mutative)

    Adds a Card to players cards

11. public void set_mpaza(Sullogi s); Transformer(Mutative)

    Adds a collection to players mpaza

12. public Sullogi get_mpaza(); Accessor (Selector)

    Returns the mpaza collection

13. public Sullogi getCards(); Accessor (Selector)

    Returns the Cards collection of a player

14. public void Tichu(); Transformer(Mutative)

    Sets players choice to Tichu

15. public void GrandTichu(); Transformer(Mutative)

    Sets players choice to Grand Tichu

16. public int getChoice(); Accessor (Selector)

    Returns the choice of a player

17. public void Played(); Transformer(Mutative)

    Set hasPlayed to true

18. public void has_finished(); Transformer(Mutative)

    Set has_finished to true

19. public boolean Has_Played(); Observer

    Returns if a player has played at least one time

20. public boolean Get_has_finished(); Observer

    Returns if a player has finished the game(partida)

Package Controller

Class Controller

Αυτή η κλάση είναι ουσιαστικά το μυαλό του παιχνιδιού. Είναι υπεύθυνη για τη δημιουργία ενός νέου παιχνιδιού, μιας νέας παρτίδας , τη δημιουργία στιγμιοτύπων παικτών, ομάδων σειράς και γύρων και φυσικά τη σύνδεση μεταξύ των γραφικών και του Model.Αυτό που κάνει η κλάση αυτή είναι να παίρνει τις επιλογές του χρήστη μέσω των γραφικών και να πραγματοποιεί οποιαδήποτε ενέργεια χρειάζεται έτσι ώστε το παιχνίδι να παίζεται σωστά. Φυσικά είναι υπεύθυνη αυτή η κλάση για να υπολογίζει το σκορ και να ενημερώνει πότε τελειώνει το παιχνίδι.

Τα attributes της κλάσης:

1) private int fold, isready, points,isready2,euxi;
2) private boolean notstarted, empty_table, new_round, collectionHasPlayed, antallages_finished;
3) private Team team1,team2;
4) private Player P1,P3,P2,P4;
5) private ArrayList <Player> players;
6) private Turn turn;
7) private Round round ;
8) private Sullogi allcards, cardsToPlay,round_bomb;

Οι μέθοδοι:

1. public boolean GetCollectionHasPlayed(); Observer

   Returns if a collection has played

2. public void PlayCollection(ArrayList<Integer> cardsPosition) Transformer

   Sets the collectionHasPlayed to true if the collection played, otherwise false

3. public int seeTurn(); Accessor (Selector)

   Returns which player has the turn

4. public void set_Fold(); Transformer (Mutative)

   increase the fold variable(fold++) or sets it to 0

5. public boolean tableIsEmpty(); Observer

   Return true if the table is empty false otherwise

6. public void set_started(); Transformer (Mutative)

   Sets the variable not_started to false

7. public boolean not_started(); Observer

   Return true if the game has not started false otherwise

8. public int Get_isready(); Accessor (Selector)

   Returns the number of players

   which are ready for dealing the last 6 cards

9. public void isready(); Transformer (Mutative)

   Increases the variable isready by 1(isready++)

10. public void sort_cards();Transformer (Mutative)

    Sort each player's cards

11. public void setGrandTichu();Transformer (Mutative)

    Sets Grand Tichu for a player

12. public void setTichu();Transformer (Mutative)

    Sets Tichu for a player

13. public void setScore() ; Transformer (Mutative)

Sets the score of a game after one deal(partida) has finished

14. public int GetScore() ; Accessor(Selector)

    Returns the score of the game

15. public void euxiMeMahjong() ; Transformer (Mutative)

    Sets the wish of the player

16. public void init_player_cards() ; Transformer (Mutative)

    Initializes players cards in the beginning

17. public void init_table() ; Transformer (Mutative)

    Initializes some things(allcards,turn,round) for a new deal(partida)

18. public boolean Get_new_round() ; Observer

    Return true if we have a new round

19. public void makeChanges(ArrayList<Integer> cardsPosition) ; Transformer

    Make changes after a round started

20. public boolean partida_has_finished(); Observer

    Return true if a deal(partida) has finished, false otherwise

21. public boolean game_has_finished(); Observer

    Return true if a game has finished, false otherwise

22. public void set_Turn();Transformer (Mutative)

    Give the turn to player who has mahjong in the beginning

23. public void Setantallages(boolean b); Transformer (Mutative)

    Sets antallages_finished to true if exchanges have finished

24. public boolean antallages();Observer

    Return true if exchanges have finished false otherwise

25. public int Get_isready2(); Accessor (Selector)

Returns the number of players which are ready for exchanging

26. public void isready2();Transformer (Mutative)

    Increases the variable isready2 by 1

27. public void makeAntallages(ArrayList<Integer> cardsPosition);Transformer (Mutative)

    Makes the exchanges

28. public String[] PreviousMove() ; Accessor (Selector)

    Returns a string array of the last cards that have been played

29. public void setmpaza(); Transformer

    Sets the mpaza after a hand played

30. public String euxi() ; Accessor

    Return the wish as a String

31. public void setHundTurn() ; Transformer

    Sets the turn when hund card played

Package View

Αυτό το πακέτο αποτελείται από μία κλάση που δημιουργεί ένα frame και μέσα σε αυτό ένα panel.Μέσα σε αυτό το panel υπάρχουν 2 panels για κάθε παίχτη ,όπου στο πρώτο περιλαμβάνονται οι κάρτες του και στο δεύτερο η συλλογή που θέλει να παίξει. Επίσης υπάρχει ένα κεντρικό panel που είναι το ταμπλό του παιχνιδιού και κουμπιά για το Tichu, το GrandTichu , το Πάσο και το Παίξε. Ακόμα υπάρχουν 56 κουμπιά που θα αντιστοιχούν στις κάρτες του παιχνιδιού (πχ το κουμπί 0 θα αντιστοιχεί στην κάρτα που βρίσκεται στη θέση 0 στη συλλογή με όλες τις κάρτες).Επίσης τα 14 πρώτα κουμπιά είναι για τον παίκτη 1 τα επόμενα για τον παίκτη 2 κλπ. Όταν πατιέται μια κάρτα αν είναι η σειρά του παίκτη θα μεταφέρεται στο χώρο όπου βρίσκονται οι κάρτες που

θέλει να παίξει και αν πατήσει το κουμπί παίξε και η συλλογή είναι έγκυρη θα μεταφερθούν οι κάρτες στο κεντρικό panel (ταμπλό).Ακόμα υπάρχουν κουμπιά για το Quit και New Game και 2 χώροι μηνυμάτων , ένας κάτω από το κεντρικό ταμπλό όπου ενημερώνει με μηνύματα τους παίκτες και ένας πάνω που γράφεται το σκορ. Επίσης υπάρχουν labels αν πατηθεί Tichu η GrandTichu και ένας χώρος που γράφεται ποια ευχή έχει ζητηθεί. Επίσης υπάρχουν 2 κλάσεις που ουσιαστικά είναι Dialogs με το χρήστη για να μας ενημερώνει για τις επιλογές του.

Άρα έχουμε 3 κλάσεις:

- GraphicUI
- MenuDialog
- WishDialog

Η GraphicUI είναι υπεύθυνη για το μεγαλύτερο μέρος των γραφικών. Σε αυτή δημιουργούνται όλα τα γραφικά εκτός από τους διαλόγους, ενώ περιέχονται σε αυτή , οι κλάσεις που κάνουν extend την ActionListener , έτσι ώστε να ανανεώνεται συνέχεια το γραφικό περιβάλλον. Ακόμα περιέχει και κλάση που αναπαράγει ήχους , φαινόμενο που κάνει το παιχνίδι πιο διασκεδαστικό.

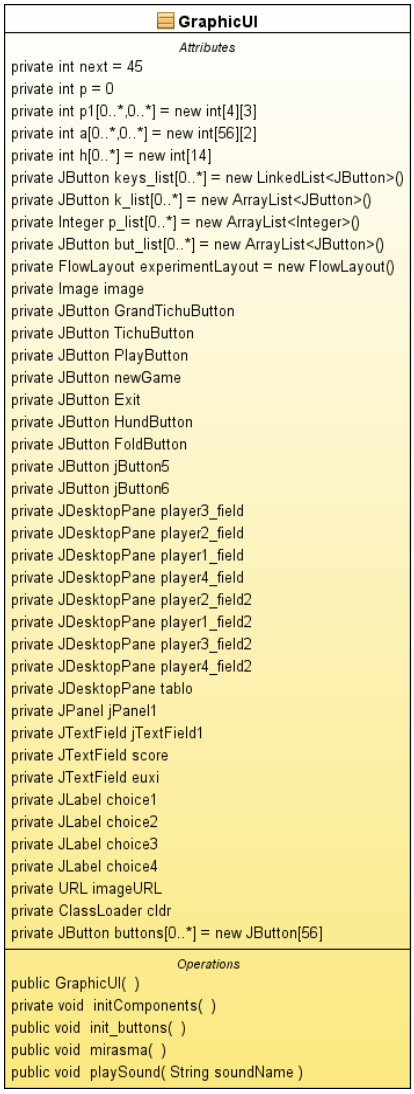
Οι άλλες 2 κλάσεις είναι υπεύθυνες για το διάλογο του παιχνιδιού με το χρήστη. Έτσι η MenuDialog χρειάζεται στις περιπτώσεις:

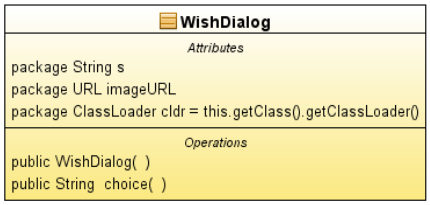
- Για το που θα δοθεί ο δράκος
- Για το ποιο χαρτί θέλει να είναι ο φοίνικας σε περίπτωση που δεν είναι προφανής(πχ σε straight 5-6-7-8 και φοίνικας αν είναι 4 ή 9)
- Για να ρωτήσει το χρήστη αν όντως θέλει να κάνει Exit
- Για να ρωτήσει το χρήστη αν όντως θέλει νέο παιχνίδι

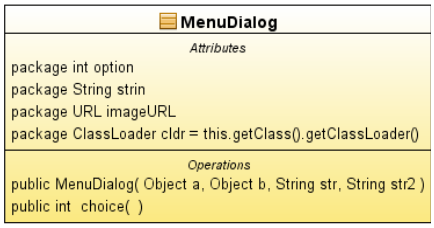
Η WishDialog από την άλλη χρειάζεται στην περίπτωση:

- Για το ποιο χαρτί θέλει να ζητήσει ο παίκτης

Μια αναπαράσταση των 3 κλάσεων σε UML.







ΤΕΛΟΣ