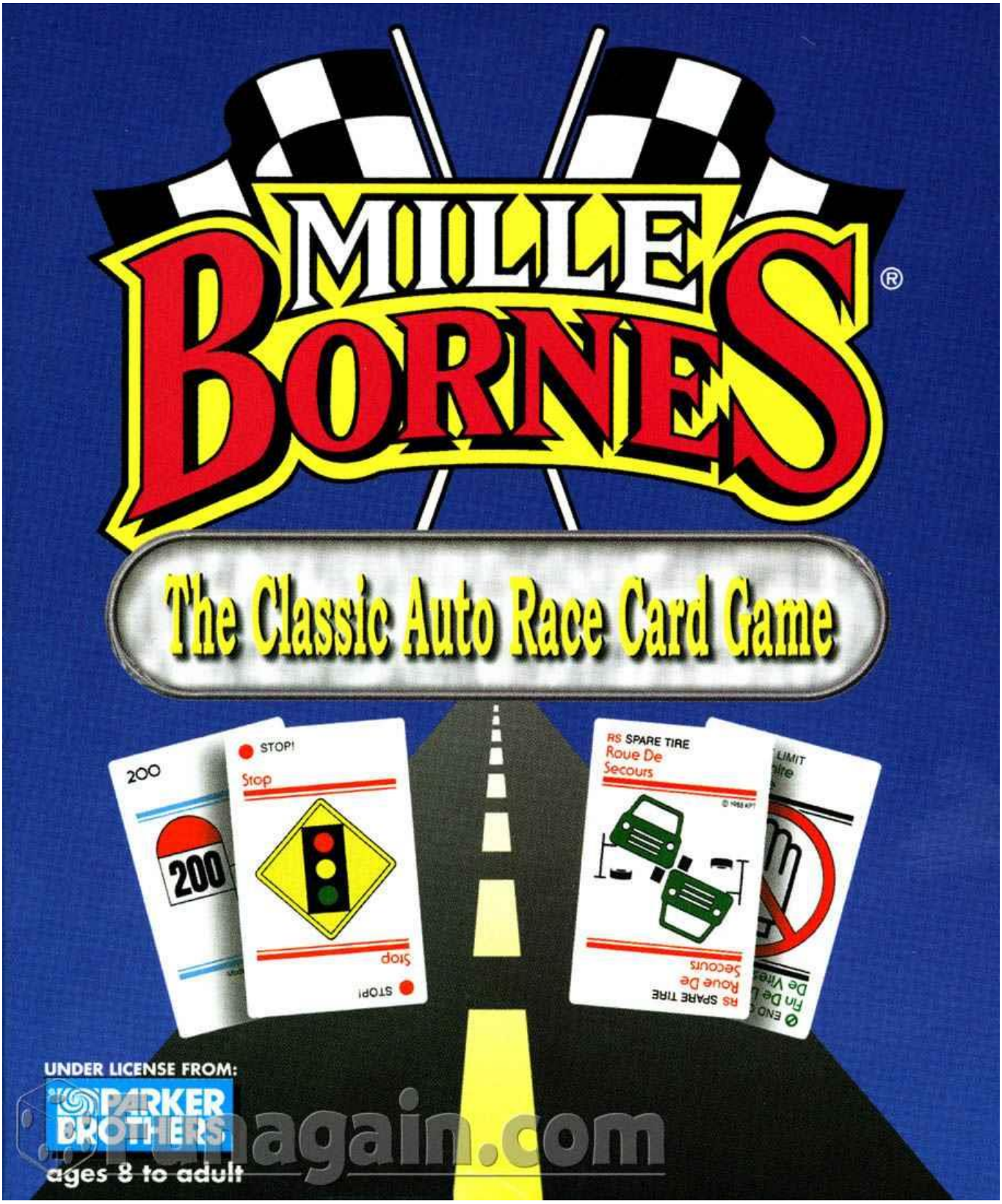
PROJECT 2014-2015



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 η διεπαφή Card, οι κλάσεις Distance , Hazard , Remedy , Safety, κλάσεις που κληρονομούν την Hazard , Remedy , Safety , η κλάση Sullogi και Player,η κλάση Turn και τέλος η κλάση Deck.

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 Distance , Hazard , Remedy , Safety

.

Το interface αυτό μας παρέχει τις εξής μεθόδους:

1. *public String getCardName(); {Accessors}*
   return String value
2. *public void setCardName(String card); {Transformer}*
   sets the card's name
3. *public abstract boolean match(Card c); {Observer}*
   Return if the card C match with the instance that called
4. *public String toString();*
   The string representation of a card

Στην συνέχεια έχουμε τις κλάσεις Distance, Hazard, Remedy, Safety που υλοποιούν την διεπαφή Card.

Class Distance

Τα attributes:

1. private int value =0; //Card value
2. private String cardName; //Card name

Οι υπόλοιπες μέθοδοι :

1. public Distance(int value) {Constructor}
   constructs a new Distance card with new value and name.
2. public int getValue() {Accessors}
   return the value of the card
3. public void setValue(int value) {Transformer}
   set new value for the card
4. public String toString()
   The string representation of a card
5. public boolean match(Card c) {Observer}
   Return if the card C match with the instance that called
6. public String getCardName() {Accessors}
   returns the card's name
7. public void setCardName(String card) {Transformer}
   card new name

EXTRA:

8. public String getCardType() {Accessors} (Αυτή χρειάζεται για να πάρω τον τύπο & όνομα της κάρτας που είναι το ίδιο.)

Class Hazard

Τα attributes:

1. *private String cardName; //Card name*
2. *private String type; //Card type*

Οι υπόλοιπες μέθοδοι :

1. public Hazard(String cardName,String type) {Constructor}
   constructs a new Hazard card with new cardName and type
2. public Hazard(String cardNameType) {Constructor}

constructs a new Hazard card with new cardName

3. public String getType() {Accessors}
   return the type of the card
4. public void setType(String type) {Transformer}
   set new type for the card
5. public String getCardName() {Accessors}
   return the name of the card
6. public void setCardName(String cardName) {Transformer}
   set new name for the card
7. public String toString()
   The string representation of a card

Class Remedy

Τα attributes:

3. *private String cardName; //Card name*
4. *private String type; //Card type*

Οι υπόλοιπες μέθοδοι :

1. public Remedy(String cardName,String type) {Constructor}

   constructs a new Remedy card with new cardName and type

2. public Remedy (String cardNameType) {Constructor}

   constructs a new Remedy card with new cardName

3. public String getType() {Accessors}

   return the type of the card

4. public void setType(String type) {Transformer}

   set new type for the card

5. public String getCardName() {Accessors}

   return the name of the card

6. public void setCardName(String cardName) {Transformer}

set new name for the card

7. public String toString()

   The string representation of a card

Class Safety

Τα attributes:

1. *private String cardName; //Card name*
2. *private String type; //Card type*

Οι υπόλοιπες μέθοδοι :

1. public Hazard(String cardName,String type) {Constructor}
   constructs a new Hazard card with new cardName and type
2. public String getType() {Accessors}
   return the type of the card
3. public void setType(String type) {Transformer}
   set new type for the card
4. public String getCardName() {Accessors}
   return the name of the card
5. public void setCardName(String cardName) {Transformer}
   set new name for the card
6. public String toString()
   The string representation of a card

Class

Accident ,FlatTire ,OutOfGas ,SpeedLimit ,Stop , EndOfLimit ,Gasoline ,Repairs, Roll, SpareTire, DrivingAce, ExtraTank, PunctureProof, RightOfWay

Αυτές οι κλάσεις κάνουν extends τις κλάσεις Hazard, Remedy, Safety αντίστοιχα για κάθε γραμμή και με την εντολή super αρχικοποιούν τις τιμές type, cardName.

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 106 κάρτες.

Τα attributes:

1. *private ArrayList<Card> cards;*

Οι υπόλοιπες μέθοδοι :

1. *public Sullogi() {Constructor}*

2. *public void init_cards() {Transformer}*
   Initializes and shuffles the 106 cards.
3. *public boolean isEmpty() {Observer}*
   Returns true if this list contains no elements.
4. *public void addCard(Card cd) {Observer}*
   Adds a card to the list.
5. *public int getValue(int target) {Accessors}*
   returns the card's value in position target
6. *public String getType(int target) {Accessors}*
   returns the card's type in position target
7. *public void removeCard(Card cd) {Transformer}*
   Removes a card from the list.
8. *public int size() {Transformer}*
   Returns the size of a list
9. *public Card getCard(int target) {Accessors}*
   returns the card in position 'target'
10. *public void clearAll() {Transformer}*
    Clears an ArrayList
11. *public ArrayList<Card> getCards() {Accessors}*

Clears an ArrayList

Class Deck

Η κλάση αυτή αναπαριστά ένα deck του κάθε παίχτη με τα διάφορα δεδομένα που αυτό μπορεί να έχει.

Τα attributes:

1. ~~private static int num=0;~~ (removed)
2. ~~private boolean canStart,new_round,win;~~ (removed)
3. private boolean firstTime=true;
4. private Player p;
5. private static Sullogi DrawPile=new Sullogi();
6. private static Sullogi DiscardPile= new Sullogi();
7. private DistancePile distanceP;
8. private BattlePile battleP;
9. private SafetyPile safetyP;
10. private SpeedPile speedP;
11. ~~private Card tempCard;~~ (removed)
12. private boolean drivingAce, extraTank, puncturePproof, rightOfWay;

Οι υπόλοιπες μέθοδοι :

1. *public Deck() {Constructor}*
   Construct a new deck and set the new player,new pilles for the player and set some variables
2. ~~*public boolean isCanStart() {Observer}*~~
   ~~*return true if the player can start playing otherwise false*~~
3. ~~*public void setCanStart(boolean canStart) {Transformer}*~~
   ~~*set true if the player must play now, false otherwise*~~
4. ~~*public boolean isNew_round() {Observer}*~~
   ~~*return true if this is a new round ,false otherwise*~~
5. ~~*public void setNew_round(boolean new_round) {Transformer}*~~
   ~~*set true if this is a new round,else false*~~
6. *public boolean isFirstTime() {Observer}*
   return true if the player never play for that game,false otherwise
7. *public boolean isWinner() {Observer}*
   return true if that player reach 1000milles and win the game, else false
8. ~~*public void setWinner() {Transformer}*~~
   ~~*set the variable win true that mean that player is the winner.*~~

9. *public Player getPlayer() {Observer}*
   return the player of the current deck
10. ~~*public void setPlayer(Player newPlayer) {Transformer}*~~
    ~~*set new player on that deck*~~
11. *public boolean isEmptyDrawPile() {Observer}*
    Return true if drawPile contains no elements,false otherwise.
12. *public static Card popDrawPile() {Observer}*
    get new card from the drawPile
13. ~~*public boolean drawPileIsEmpty() {Observer}*~~
    ~~*return true if drawPile contains no elements,false otherwise*~~
14. *public static void pushDiscardPile(Card cd) {Transformer}*
    adding a Card to the discardPile
15. *public Card getTempCard() {Observer}*
    getting the 7th card for the player
16. ~~*public void setTempCard(Card tempCard) {Transformer}*~~
    ~~*set new temp card*~~
17. *public void pushDistancePile(Card cd) {Transformer}*
    adding a Card to the distancePile
18. *public void pushBattlePile(Card cd) {Transformer}*
    adding a Card to the battlePile
19. *public void pushSafetyPile(Card cd) {Transformer}*
    adding a Card to the safetyPile
20. *public void pushSpeedPile(Card cd) {Transformer}*
    adding a Card to the speedPile

EXTRAS:

1. public void setBattlePileDeck(Deck d) {Transformer}
   set on that deck the enemy deck
2. public void setFirstTime(boolean x) {Transformer}
   set new value on the firstTime
3. public void setDrivingAce(boolean drivingAce) {Transformer}
   setting new value on variable drivingAce
4. public void setExtraTank(boolean extraTank) {Transformer}
   setting new value on variable extraTank
5. public void setPunctureProof(boolean punctureProof) {Transformer}
   setting new value on variable punctureProof
6. public void setRightOfWay(boolean rightOfWay) {Transformer}
   setting new value on variable rightOfWay

7. public DistancePile getDistancePile() {Observer}
   getting the distance Pile
8. public BattlePile getBattlePile(){Observer}
   getting the battle Pile
9. public SafetyPile getSafetyPile(){Observer}
   getting the Safety Pile
10. public SpeedPile getSpeedPile(){Observer}
    getting the Speed Pile
11. public Sullogi getDrawPile() {Observer}
    getting the Draw Pile
12. public boolean isDrivingAce(){Observer}
    returning the driving ace value
13. public boolean isExtraTank(){Observer}
    returning the Extra tank value
14. public boolean isPunctureProof(){Observer}
    returning the puncture proof value
15. public boolean isRightOfWay(){Observer}
    returning the right of way value

Class BattlePile

Στην κλάση αυτή αποθηκεύετε η κάρτα επίθεσης που έχει παίχτη στον παίχτη του deck.

Τα attributes:

1. private Card card=null;
2. private Deck d;

Ο ι υπόλοιπες μέθοδοι :

1. *public void addCard(Card newCard) {Transformer}*
   change the value of variable card to the newCard
2. ~~*public boolean thereIsAnyCard() {Observer}*~~
   ~~*Return true if card!=null, false otherwise.*~~
3. ~~*public boolean canAddCard(Card card) {Observer}*~~
   ~~*Return true if the variable card can be added to battlePile,false otherwise*~~

EXTRAS:

1. public BattlePile(Deck d) {Transformer}
   Setting the enemy deck

2. public void setCard(Card cd) {Transformer}
   change the value of the card to new
3. public Card getCard() {Observer}
   getting the card of the instance
4. public void removeCard() {Transformer}
   set the card of the class to null

Class DistancePile

Στην κλάση αυτή αποθηκεύονται οι κάρτα απόστασης που έχει παίξει ο παίχτης.

Τα attributes:

1. *private Sullogi card;*
2. ~~*private boolean SpeedLimited=false;*~~
3. *private int totalMilles;*
4. *private int counter=0;*
5. *private Deck d;*

Οι υπόλοιπες μέθοδοι :

1. *void addCard(Card cd) {Transformer}*
   add new card to the card Collection
2. *boolean CanMove() {Observer}*
   return true if the player can move(if can add a distance card to the deck),false otherwise
3. *boolean isSpeedLimited() {Observer}*
   r eturn true if there is any speed limit for that deck,false otherwise
4. ~~*public void setSpeedLimited(boolean SpeedLimited) {Transformer}*~~
   ~~*set new value for the variable speedLimited*~~
5. *public boolean isTheWinner() {Observer}*
   return true if the player reach the 1000miles, false otherwise

EXTRAS:

1. *public int getTotalMilles() {Observer}*
   Returning the total miles
2. *public Card getLastCard() {Accessors}*
   returning the last card of Array if it's not empty

Class SafetyPile

Στην κλάση αυτή αποθηκεύονται οι κάρτες ασφαλείας του παίχτη.

Τα attributes:

1. private Sullogi card;
2. private Deck d;

Ο ι υπόλοιπες μέθοδοι :

1. *public void addCard(Sullogi card) {Transformer}*
   add new card to the card Collection

EXTRAS:

1. *public SafetyPile(Deck d) {Transformer}*
   initialize the deck
2. *public boolean isEmpty(){Observer}*
   Return if the card list is empty
3. *public Sullogi getAllCard(){Accessors}*
   Return all the array with cards

Class SpeedPile

Στην κλάση αυτή αποθηκεύετε οι κάρτες ταχύτητας που έχουν παιχτεί για αυτόν τον παίχτη.

Τα attributes:

1. private Sullogi card;
2. private Deck d;

Ο ι υπόλοιπες μέθοδοι :

1. *public void addCard(Sullogi card) {Transformer}*
   add new card to the card Collection

EXTRAS:

1. *public SpeedPile(Deck d) {Transformer}*
   initialize the deck
2. *public Card getCard() {Accessors}*
   return the card of the instance

Class Player

Η κλάση αυτή αναπαριστά ένα παίκτη μέσα στο παιχνίδι.

Τα attributes:

1. *private String name;*
2. *private Sullogi cards;*
3. *private int ID;*
4. ~~*private boolean finished;*~~

Οι υπόλοιπες μέθοδοι :

1. public Player(String name, int ID) {Constructor}
   Constructs a new Player with the given parameter name and ID.
2. public String getName() {Accessors}
   Returns the name of the player
3. ~~public void setName(String newName) {Transformer}~~
   ~~sets the name of the player to newName~~
4. public Sullogi getCards() {Accessors}
   Returns the cards that player have on his hands
5. public void setCards(Sullogi cards) {Transformer}
   adds a Card to players cards
6. public int getID() {Accessors}
   Returns the ID of a player.
7. public void setID(int ID) {Transformer}
   It sets the ID of a player
8. ~~public boolean isFinished() {Observer}~~
   ~~return if a player has finished for that round~~
9. ~~public void setFinished(boolean finished) {Transformer}~~
   ~~sets the variable finished to true or false depends on call.~~

EXTRAS:

1. public Card removeCard(String type) {Observer}
   return if the card with (type):type founded and removed successful
2. public Card getCard(String type) {Observer}
   return if the card with type exist on player cards

Class Turn

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

Τα attributes:

1. private int currentID;

Ο ι υπόλοιπες μέθοδοι :

1. public Turn() {Constructor}
2. public void setID(int id){Transformer}
   Sets the player turn.(which player has the turn to play)
3. public int getID(){Accessors}
   returns the player ID whose turn is to play
4. ~~public boolean checkIfPlayerFinished(Player P){Observer}~~
   ~~Checks if a player has finished~~

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 deck, 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:

1. private Deck d1;
2. private Deck d2;
3. private Player p1;
4. private Player p2;
5. private Turn turn;
6. ~~private Sullogi allcards = new Sullogi();~~
7. ~~private boolean cardValid;~~

Ο ι υπόλοιπες μέθοδοι :

1. public boolean PlayCard(Card cd) {Transformer}

take an card as input, the card that player want to play. Checking if that card match and can be played.

2. public Card getTempCard() {Accessors}
   return the temp card that some1 player get from the drawPille
3. public void shareCards() {Transformer}
   we sharing 6 card from the 106 to the player
4. public int seeTurn() {Accessors}
   Returns which player has the turn
5. public int game_has_finished() {Observer}
   Return true if a game(a player reaches 1000 miles) has finished, false otherwise

EXTRAS:

1. public void DiscardCard() {Observer}
   Found and delete the card with type:type from the player card
2. public void nextPlayer() {Transformer}
   set new Id on the Turn, depends on previous id
3. public int p1Milles() {Accessors}
   getting the total Miles of the player1
4. public int p2Milles() { Accessors }
   getting the total Miles of the player2
5. public Deck getDeck1() { Accessors}
   getting the Deck 1
6. public Deck getDeck2() { Accessors}
   getting the Deck 2

Package View

Αυτό το πακέτο θα αποτελείται από μία κλάση που θα δημιουργεί ένα ένα frame και μέσα σε αυτό 3 panels,1 panel θα έχει κάθε παίχτης και 1 panel θα είναι στην μέση όπου μέσα σε αυτό θα υπάρχουν τα DiscardPille,DrawPille και οι πόντοι των κάθε παιχτών. Επίσης 30 κουμπία 14 για κάθε παίχτη και 2 στην μέση. Στην κορυφή θα αναγράφετε η σειρά του κάθε παίχτη.

Class GraphicUI

Τα attributes:

private Controller game;
private Image image;
private myDesktopPane basic_panel;
private JButton Exit, newGame;
private JDesktopPane deck1;
private JDesktopPane deck2, tablo;
private JTextField turn, p1Milles, p2Milles, drawPnum;
private JButton[] buttons = new JButton[30];
private URL imageURL;
private ClassLoader cldr;

Οι υπόλοιπες μέθοδοι :

1. public GraphicUI() {Constructor}
   Creates a new Window and initializes some buttons and panels
   postconditions: Creates a new Window and initializes some buttonsand panels starting a new game.
2. private void initComponents() { transformer}
   initialize some buttons and labels
3. public void init_buttons(){ transformer}
   sets some buttons and labels for a new deal
4. public void cardShare(){ transformer}
   sharing the cards to the players and init the buttons
5. public class myDesktopPane extends JDesktopPane
   a class which is used for putting a background image to a jdesktoppane
6. private class SettingsListener implements ActionListener { transformer}
   doing some action after New Game or Exit button has been pushed
7. private class CardListener implements ActionListener { transformer}
   doing some action after a card button has been pushed
8. private class RightCardListener extends MouseAdapter { transformer}
   doing some action after a right push on a button

Class MenuDialog

Τα attributes:

1. int option;
2. String strin;
3. URL imageURL;
4. ClassLoader cldr = this.getClass().getClassLoader();

Οι υπόλοιπες μέθοδοι :

1. public MenuDialog(Object a, Object b, String str, String str2) {Constructor}
   Creates a new Menu Window
2. public int choice() {Accessors}
   Returns the choice of a player

FINITO!!!