ΗΥ-252 Αντικειμενοστρεφής Προγραμματισμός

Προγραμματιστική Εργασία Χειμερινού Εξαμήνου 2014

Mille Bornes

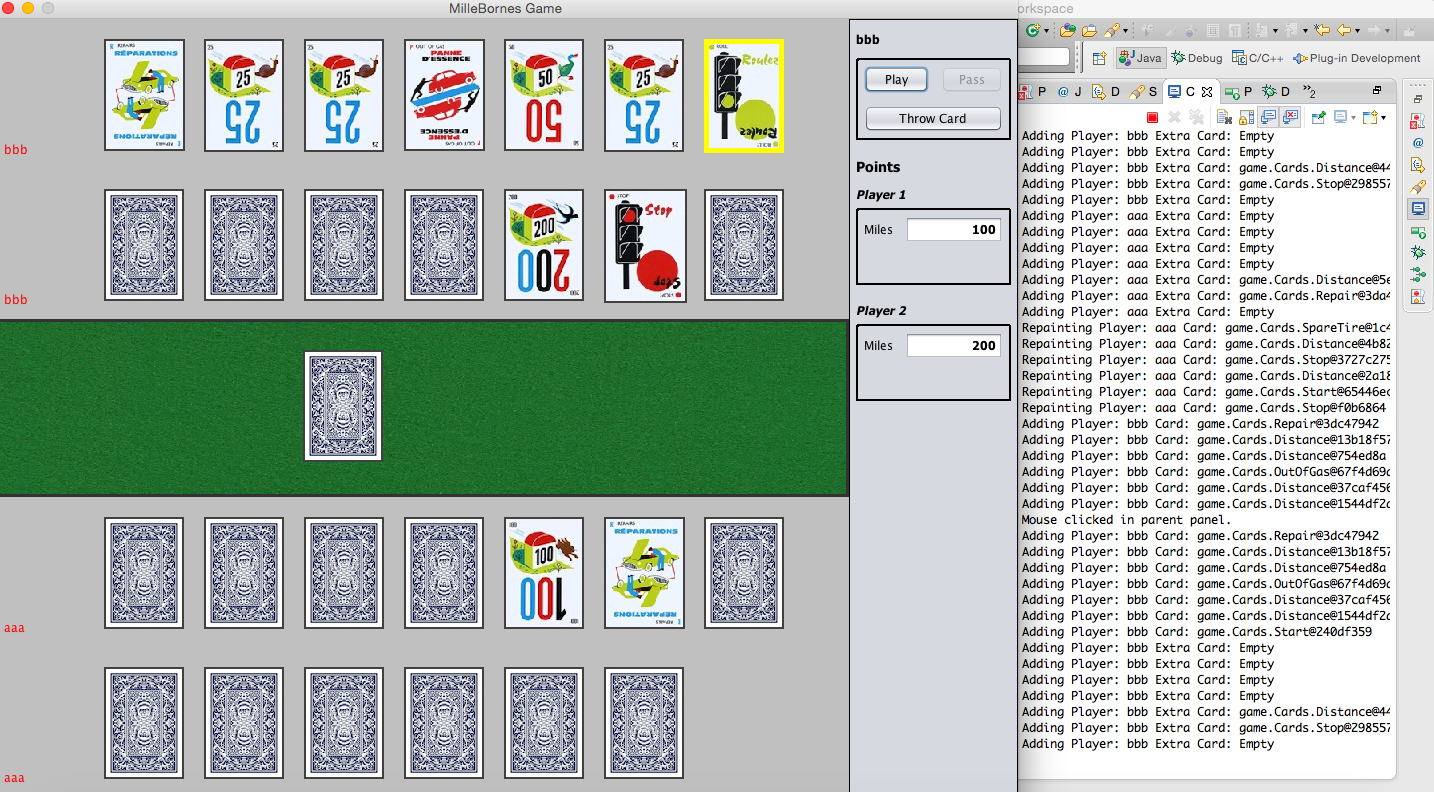


Figure : Game View

Παράδοση 2ης Φάσης

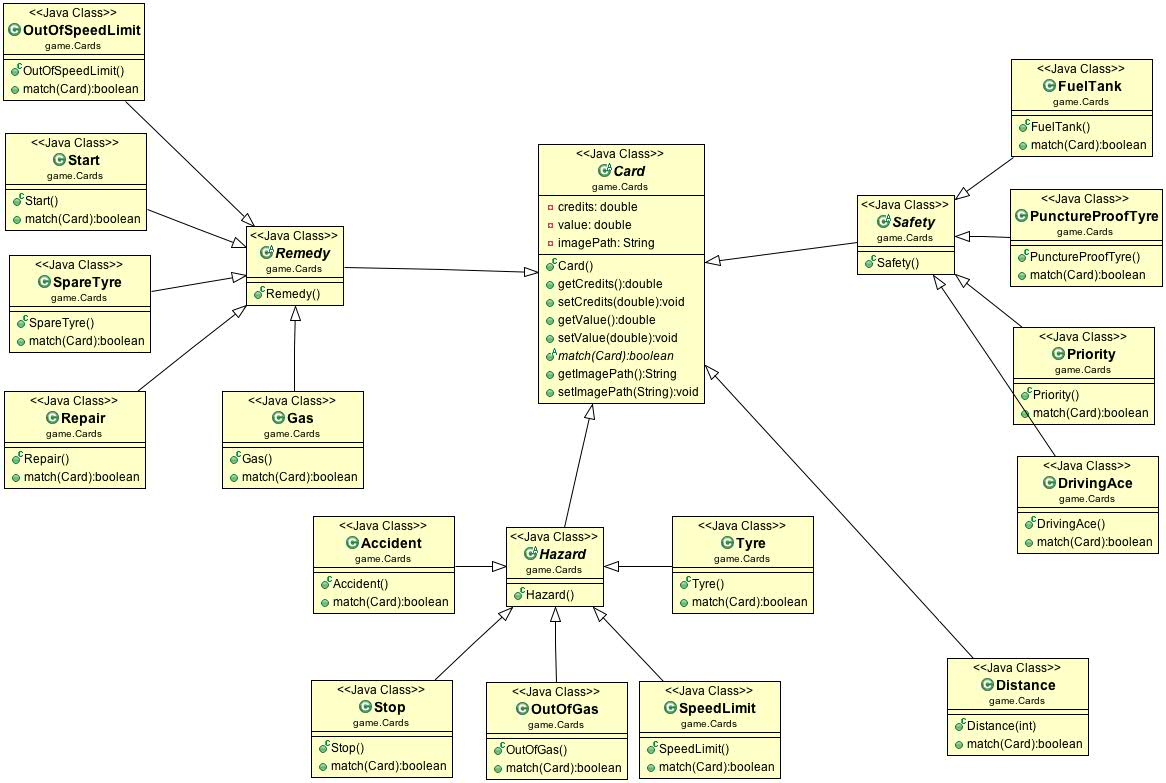
Για την υλοποίηση του παιχνιδιού έγιναν αρκετές προσθήκες κυρίως για να είναι πιο λειτουργική η υλοποίηση μας. Η κληρονομικότητα έμεινε ως είχε στην κλάση Cards και σε αυτές που την κληρονομούν. 

Figure 2: Card

Από τα κοινα χαραστηριστικά των καρτών είναι η αξία της κάρτας (value) που χρειάζεται και για την υλοποίηση της κάρτας Distance. Για παράδειγμα, όταν ο ενας παίκτης έχει οριο ταχύτητας, η τιμή αυτή θα μας βοηθήσει να καταλάβουμε αν ενας παίχτης μπορεί να ρίξει την συγκεκριμένη Distance card η οχι. Να σημειωθεί ότι η κλάσεις Safety, Remedy και Hazard είναι αφηρημένες διότι δεν υπάρχει λόγος αυτόνομης ύπαρξης επιτρέποντας με αυτό τον τρόπο την ευκολότερη διαχείριση και αναγνώριση οποιασδήποτε ειδικής κάρτας από τον πυρήνα του παιχνίδιού.

Άλλο ένα κοινό χαρακτηριστικό όλων των καρτών είναι το image path το οποίο ορίζει που βρίσκεται αποθηκευμένη η εικόνα που αντιπροσωπεύει την κάθε κράτα και θα μας βοηθήσει στην αποτύπωση τους από το View component.

Οι συλλογές των καρτών έμειναν όπως είχαν σχεδιαστεί στην πρώτη φάση. Οι συλλογές αυτές εξειδικεύουν την AbastractPile που είναι αφηρημένη κλάση και βοηθάει να ομαδοποιήσουμε τα κοινά χαρακτηριστικά των Piles.

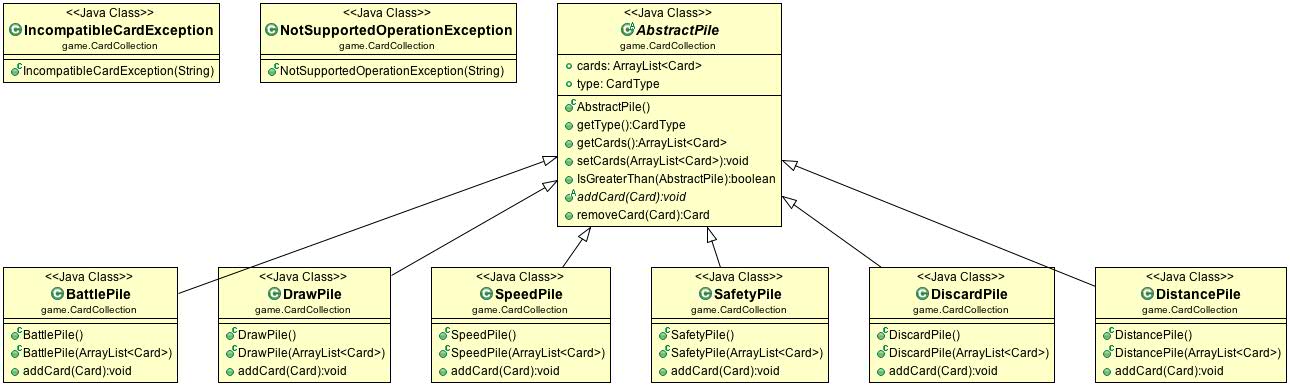


Figure 3: Card Pile

Στην κλασση του παίχτη εγιναν μικρές αλλαγές κυρίως για να μπορούμε να ξέρουμε αν εχει ρίξει καρτα εκκίνησης η αν εχει κάποια καρτα που δεν τον αφηνει να κινηθεί. Τα βασικά γνωρίσματα ενός παίχτη (πέρα απο το όνομα του) είναι το φύλλο που τράβηξε, τα μίλια που εχει διανύσει και οι τέσσερις στοίβες που τον αναγνωρίζουν (battle, speed, distance, safety) και κάποια επίσης γνώρισματα που χρησιμεύουν στον πυρήνα του παιχνιδιού οπως αν τραβήξει καρτα για αυτον τον γυρο παιχνιδιού.

|  |
| --- |
| /\*\* The name. \*/  **private** String name;    /\*\* The Miles run. \*/  **private** **int** MilesRun;    /\*\* The hand. \*/  **private** ArrayList<Card> hand;  **private** AbstractPile battle;  **private** AbstractPile speed;  **private** AbstractPile distance;  **private** AbstractPile safety;      /\*\* The has started. \*/  **private** **boolean** hasStarted;    /\*\* The state. \*/  **private** **boolean** canMove;    /\*\* The player has drawn a card in this round! \*/  **private** **boolean** hasDrawn; |

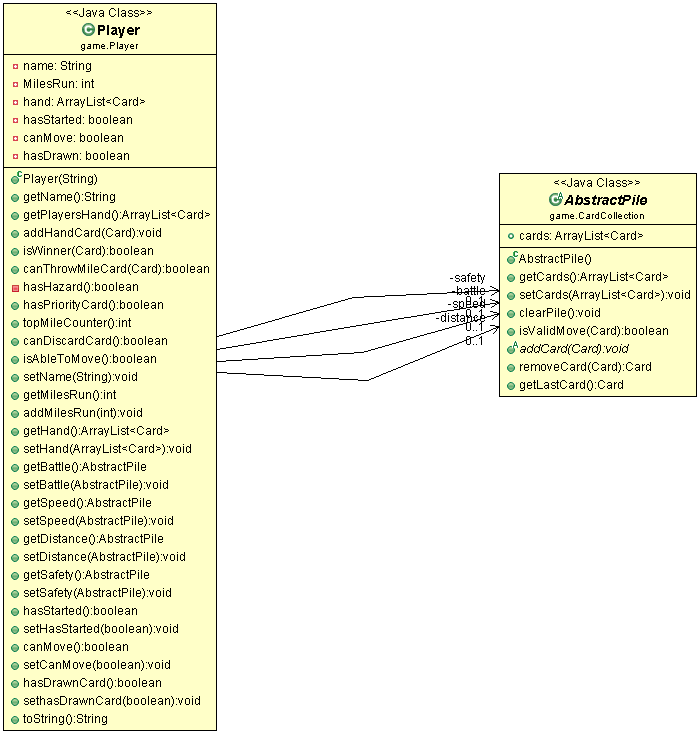


Figure 4: Player

Στην κλάση Deck η οποία προσομοιώνει το ταμπλό του παιχνιδιού δεν έγιναν κάποιες αλλαγές αφου δεν κρίθηκε αναγκαίο. Το ταμπλό αυτο περιέχει την στοίβα για να τραβάνε κάρτες οι παίχτες (DrawPile) και την στοίβα για να απορρίπτουν κάρτες οι παίχτες (DiscardPile). Επίσης περιέχει και στιγμιότυπα των δυο παιχτών (Player) και μέσω αυτων μπορεί να εχει πρόσβαση σε ολα τα πεδία της κλάσης του κάθε παίχτη. Το διάγραμμα UML αυτής της κλάσης φαίνεται παρακάτω.

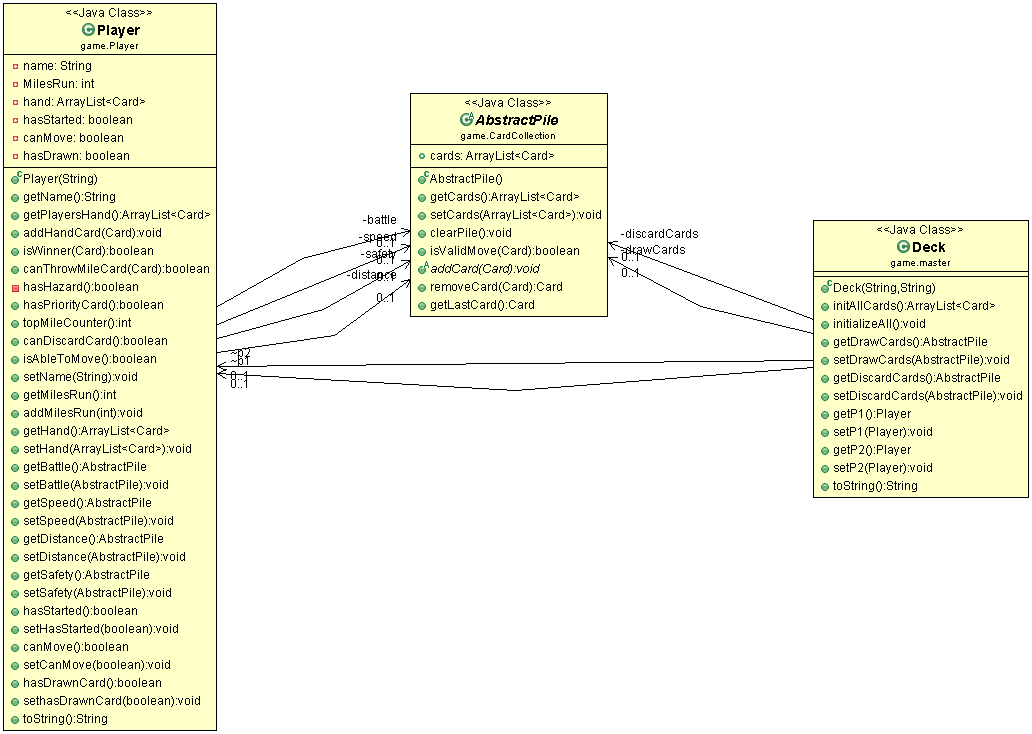


Figure : Deck

Ο πυρήνας του παιχνιδιού (GameMaster) είναι υπεύθυνος για την διεξαγωγή του παιχνιδιού από την έναρξη μέχρι και την λήξη του. Είναι επίσης ο controller αναλαμβάνοντας τον ρόλο ενδιάμεσης επικοινωνίας μεταξύ του View και του Model υιοθετόντας με αυτό τον τρόπο το design pattern MVC.

Σε αυτή την κλάση έγινε το μεγαλύτερο μέρος της υλοποίησης του παιχνιδιού αφού είναι υπεύθυνη για την σωστή διεξαγωγή του παιχνιδιού και την εφαρμογή των κανόνων. Για παράδειγμα οι μέθοδος playerSubmitsCard ελεγχει αν ο παίχτης είναι σε θέση να ρίξει αυτή την καρτα και αν ναι σε ποιο CardPile ανήκει αυτή. Επίσης σε περίπτωση που είναι Hazard Card πρέπει να ελέγξει αν ο αντίπαλος παίχτης μπορεί να δεχθεί αυτήν την επίθεση. Αντίθετα αν είναι Safety Card πρέπει να αναιρέσει την αντίστοιχη επίθεση που εχει δεχθεί. Αντίστοιχα η μέθοδος playerDiscardsCard() ελεγχει αν ο παιχτης εχει τραβηξει καρτα και αν εχει το δικαιωμα να πεταξει την συγκεκριμενη καρτα.

Να σημειωθεί οτι σε αυτούς τους ελέγχους εχει γινει εκτενής χρήση της πολυμορφικης μεθόδου match των καρτών για να γίνει έλεγχος ποτε μια καρτα μπορεί να παιχτεί πανω σε κάποια αλλη.

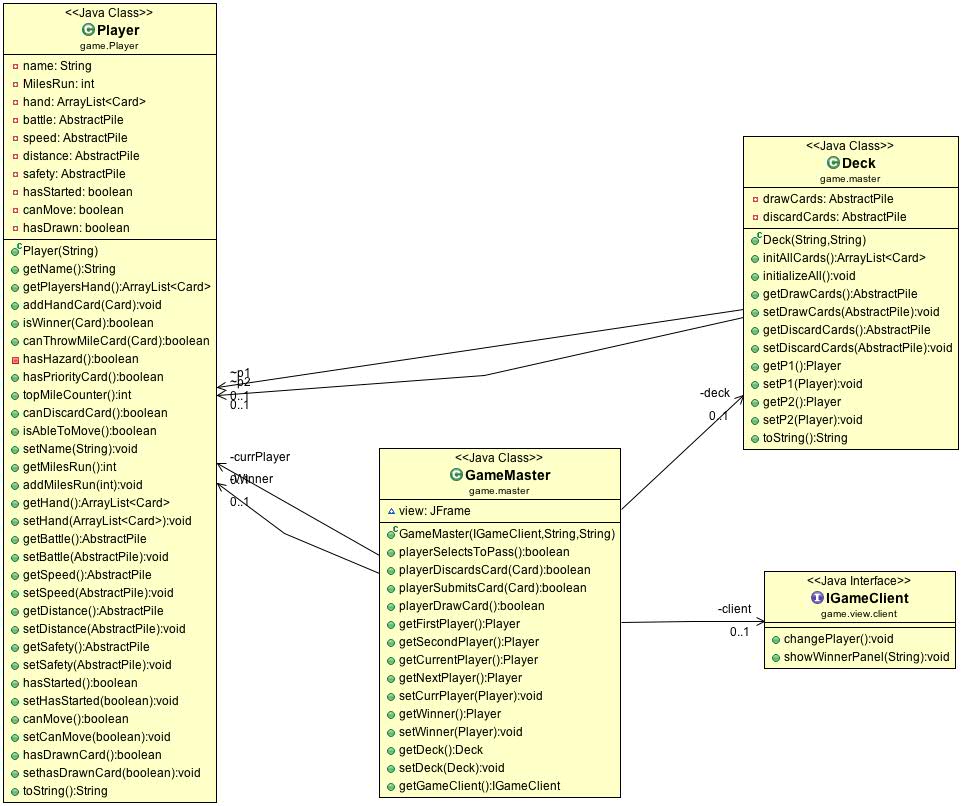


Figure 6: GameMaster

Για το GUI που δεν είχαμε περιγράψει στην πρώτη φάση υλοποιήσαμε τα εξής:

InputPlayer JDialog στο οποίο συμπληρώνουμε τα ονόματα των παιχτών.

MainGameFrame : To main frame του παιχνιδιού. Περιέχει 2 JPanels το PlayersPanel το οποίο αποτελείται εσωτερικά από τα 4 JPanels των παιχτών και το κεντρικό JPanel με τις κάρτες που ρίχνει ο κάθε παίχτης (CenterPanel) . Τα 4 JPanels των παιχτών είναι 2 HorizontalViewCardPanel για καθε παιχτη. Επιλέξαμε να ειναι δυο γιατι στην πορεία φανηκε να ειναι πιο απλο να εχουμε ενα JLabel για της κάρτες που εχει ο παίχτης στο χερι του και ενα διαφορετικο JLabel για τα Piles. Ο βασικότερος λόγος ειναι οτι αυτα τα πανελς γίνονται repaint σε διαφορεικα σημεία του παιχνιδιού. Το άλλο Panel το οποίο περιέχεται στο GameFrame είναι το GameInfoPanel που περιέχει τα Controls του παιχνιδιου και απεικονίζει τους πόντους του καθε παίχτη. Το GameFrame έχει BorderLayout διάταξη.

Για την εμφάνιση της κάρτας έχουμε την κλάση ViewCard η οποία κάνει extend την κλάση JLabel και ανάλογα με το είδος της κάρτας ζωγραφίζει στο JLabel την κατάλληλη εικόνα.

Το PlayersPanel έχει BorderLayout διάταξη. Το HorizontalViewCardPanel περιέχει τις κάρτες που έχει στο χέρι του ο κάθε παίχτης. Οι κάρτες γίνονται add στο πάνελ και γινονται καθε φορα repaint για να εμφανιστουν οι αλλαγες στον χρήστη. Κάθε φορά εμφανίζονται μόνο οι κάρτες του παίχτη του οποίου είναι η σειρά του να παίξει.

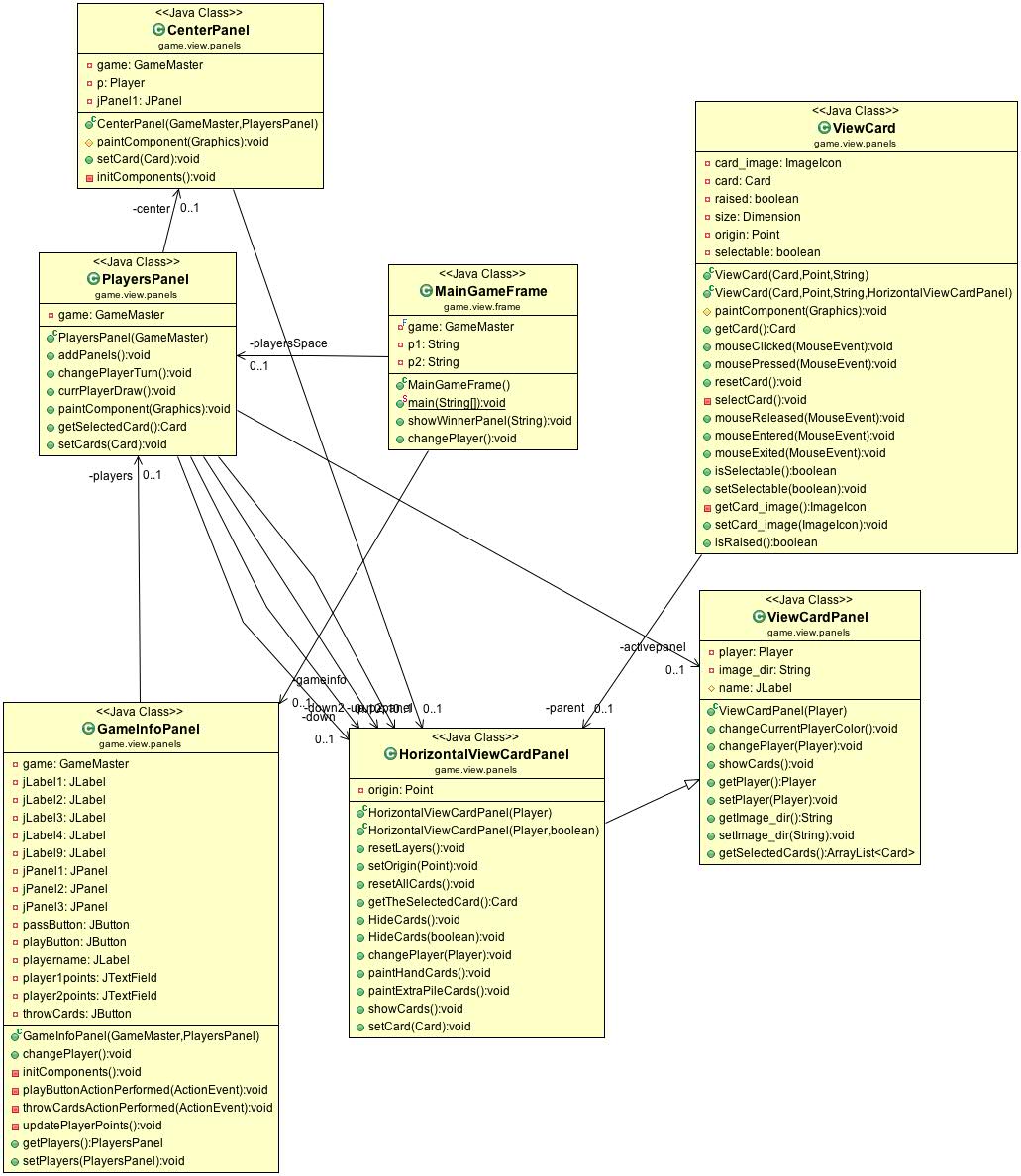


Figure 7: View

Για το πρότζεκτ επίσης δημιουργήθηκαν JUnits για να ελεγχθεί η λειτουργικότητα των βασικότερων κλάσεων (Player, Deck, GameMaster). Επίσης για να είναι το παιχνίδι μας οσο το δυνατόν πιο portable γίνεται, έχει γίνει export σε Jar και μπορεί να τρέξει σε οποιαδήποτε πλατφόρμα τρέχοντας την εντολή **java –jar MilleBornes.jar** !