Πανεπιστήμιο Κρήτης –Τμήμα Επιστήμης Υπολογιστών

HY252– Αντικειμενοστρεφής Προγραμματισμός

Διδάσκων: Ι. Τζίτζικας

Χειμερινό Εξάμηνο 2019-2020

PHASE A

SORRY! GAME

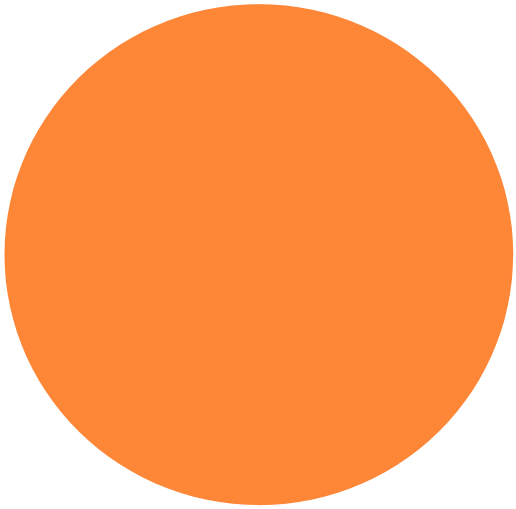
[Μιχάλης Τουτουδάκης]

[csd4054]

[7/12/2019]

Περιεχόμενα

1. Εισαγωγή....1
2. Η Σχεδίαση και οι Κλάσεις του Πακέτου Model....1
3. Η Σχεδίαση και οι Κλάσεις του Πακέτου Controller....1
4. Η Σχεδίαση και οι Κλάσεις του Πακέτου View....2
5. Η Αλληλεπίδραση μεταξύ των κλάσεων – Διαγράμματα UML....2
6. Λειτουργικότητα (Β Φάση)....2
7. Συμπεράσματα....2



1. Εισαγωγή

Εδώ θα περιγράψετε σε γενικές γραμμές ποιο μοντέλο χρησιμοποιήσατε για την εργασία σας (MVC) και θα αναφέρετε πολύ συνοπτικά τι περιέχουν οι υπόλοιπες ενότητες της αναφοράς.

Η εργασία βασίζεται στο μοντέλο MVC(Model View Controller). Το Model αναπαριστά τα objects που έχουν τα δεδομένα που χρειάζονται για την λειτουργία του προγράμματος. Το View αναπαριστά το οπτικό κομμάτι(Graphic User Interface) των περιεχομένων του Model. Το Controller συνδέεται με το Model και το View και ελέγχει την ροή των δεδομένων στο Model και ανανεώνει το View όταν αλλάζουν τα δεδομένα και κρατάει το View και Model ξεχωριστά.

2. Η Σχεδίαση και οι Κλάσεις του Πακέτου Model

Εδώ θα περιγράψετε το σχέδιο υλοποίησης της προγραμματιστικής εργασίας του πακέτου model. Συγκεκριμένα. στην Α φάση του Project θα περιγράψετε τη σχεδίαση κάθε κλάσης (π.χ., αν χωρίσατε μία κλάση σε υποκλάσεις και για ποιους λόγους), ποιες μεθόδους σκοπεύετε να χρησιμοποιήσετε στη Β φάση του Project (και για ποιο λόγο είναι χρήσιμες, ποια είναι η λειτουργικότητα τους). Μπορείτε να το χωρίσετε και σε υπό ενότητες την ενότητα αυτή (πχ ενότητα για την κλάση και τις υποκλάσεις της Card, ενότητα για την κλάση Player κλπ.). Επιπλέον, μπορείτε να συμπεριλάβετε διαγράμματα UML για να σας βοηθήσουν στην επεξήγηση των κλάσεων. Στη Β φάση του project, συμπληρώνετε τις λεπτομέρειες της υλοποίησης (πχ αλγόριθμοι που χρησιμοποιήθηκαν) και τυχόν αλλαγές.

Package Model

Σε αυτό το πακέτο περιέχονται :

Abstract class Card την οποία κάνουν extend οι κλάσεις NumberCard και SorryCard. Την NumberCard κάνουν extend οι κλάσεις SimpleNumberCard , OneNumberCard , TwoNumberCard , FourNumberCard , NumberSevenCard , NumberTenCard , NumberElevenCard.

Abstract class Square την οποία κάνουν extend οι κλάσεις StartSquare , SafetySquare , HomeSquare , SimpleSquare , SlideSquare(Την κανουν extend οι StartSlideSquare , InternalSlideSquare , EndSlideSquare).

Επίσης έχουμε τις κλάσεις class Pawn(Αναπαριστά ένα πιόνι) ,class Player(Αναπαριστά έναν παίχτη) , class Deck(Αναπαριστά το ταμπλό του παιχνιδιού με τα πιόνια , κουτάκια , κάρτες , παίχτες)

Abstract class Card and subclasses

Η abstract class Card έχει τα εξής attributes:

α) private String description;(Description of the card)

b) private boolean isPlayed; (Card is played or not)

c) private int value;(The value of the card)

Η abstract class Card παρέχει τις εξής μεθόδους :

α) public void setPlayed(boolean isPlayed) **Transformer(mutative)**

Sets the Card as played or not.

b) public boolean getPlayed() **Accessor(Observer)**

Returns if a Card is played or not.

c) public void setDescription (String description) Transformer(mutative)

Sets the description of the Card.

d) public String getDescription() Accessor(selector)

Returns the description of the Card.

e) public void movePawn(Pawn p , Deck b)

Moves a pawn with with the card value if possible

f) public void movePawn(Deck b)

Moves a pawn with with the card value if possible(Used as a special movePawn for Card7, Card11, SorryCard).

Έπειτα έχουμε τις NumberCard και SorryCard που κάνουν extend την Card.

Class SorryCard

Αυτή η κλάση αναφέρεται στις κάρτες Sorry. Τα υπόλοιπα attributes και μεθόδους που έχει η κλάση(Εκτός αυτών που κληρονομεί από την Card)

Methods:

a) public SorryCard() Constructor

Creates a sorry card with description and isPlayed = false

Επίσης η κλάση υλοποιεί τις μεθόδους: public int movePawn(Deck b) , public void setValue(int value) , public int getValue() που κληρονομεί από την abstract Card.

Αυτή η κλάση αναφέρεται στις κάρτες Sorry.

Class NumberCard

Αυτή η κλάση αναφέρεται στις κάρτες που έχουν αριθμό. Τα υπόλοιπα attributes και μεθόδους που έχει αυτή η κλάση(εκτός αυτών που κληρονομεί από την Card).

Attributes:

α**) private int value; (The number that represents what card it is)**

Method:

a) public void setValue(int value)

Sets the value of the card.

b) public int getValue()

Returns the value of the card.

Sub classes of NumberCard

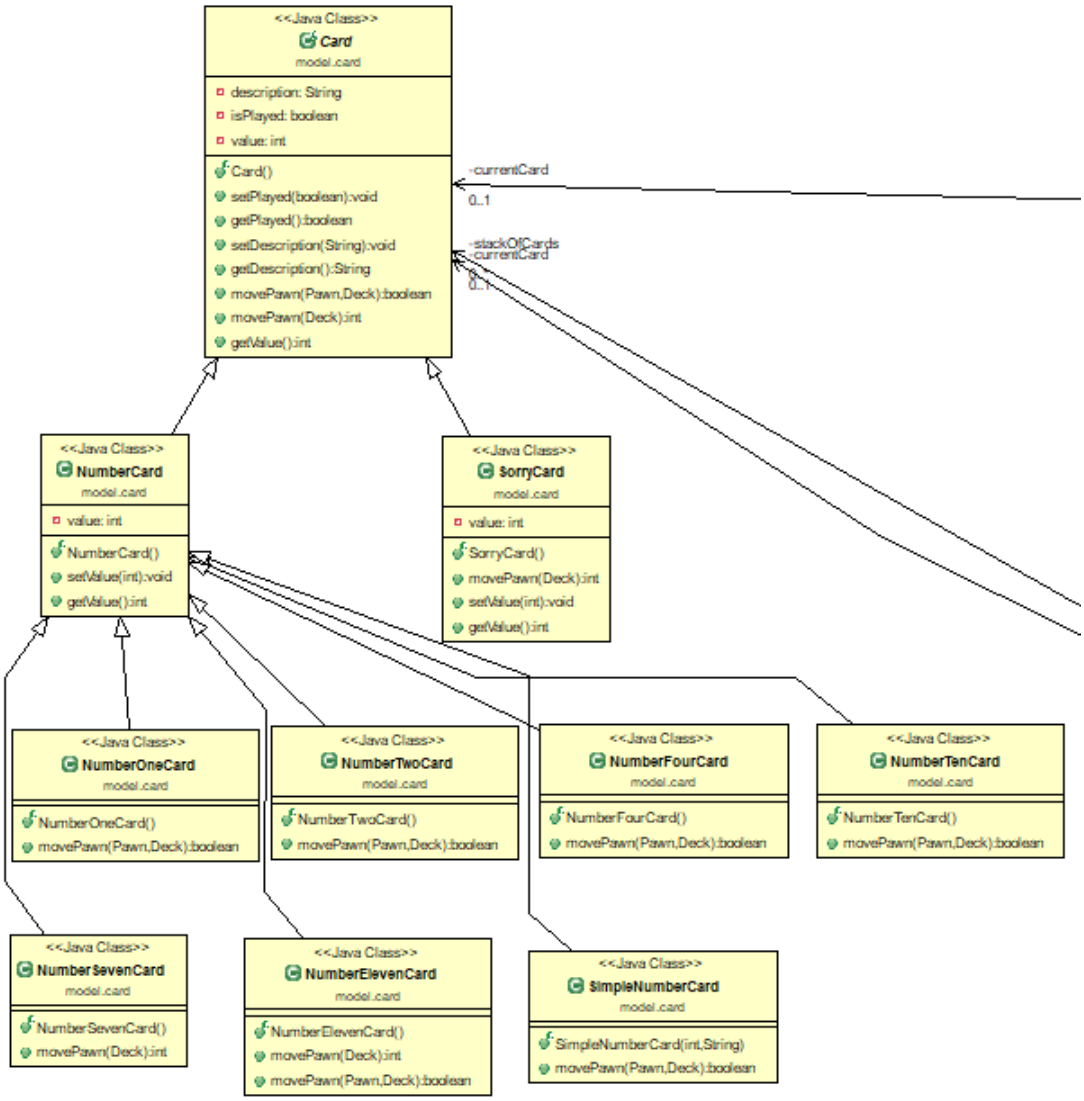
SimpleNumberCard , NumberOneCard , NumberTwoCard, NumberFourCard , NumberSevenCard , NumberTenCard , NumberElevenCard.

Αυτές οι κλάσεις όλες κάνουν extend την κλάση NumberCard. Με την χρήση του contructor αρχικοποιούν τα πεδία description και value με τις setDescription και setValue. Κάθε κάρτα έχει value και description ανάλογο.

Οι κλάσεις αυτές υλοποιούν την public boolean movePawn(Pawn p , Deck b) : **NumberOneCard , NumberTwoCard , NumberFourCard , NumberTenCard , NumberElevenCard , SimpleNumberCard**

Οι κλάσεις αυτές υλοποιούν την public int movePawn(Deck b): **NumberElevenCard , NumberSevenCard**

Αναπαράσταση των κλάσεων με UML:



Abstract class Square

Η κλάση αυτή θα είναι υπεύθυνη για τα κουτάκια που είναι στο ταμπλό. Μέσω αυτής της κλάσης κατασκευάζονται τα κουτάκια ανάλογα με τις ιδιότητες του κάθε ενός.

Attributes:

a) private boolean empty (has the bool value wheter a square is empty or not).

b) private String color (Contains the color of the square).

c) private final int posX , posY (Coordinates of the square).

d) private Pawn pawn; (The pawn that sits on the square)

Methods;

a) public Square(int posX, int posY, String color) Constructor

Creates a Square and sets coordinates, color , empty = false;

b) public void setColor(String color) Transformer(mutative)

Sets the color for the Square

c) public String getColor() Accessor(selector)

gets the color of the Square

d) public int getPosX() Accessor(selector)

Gets the X coordinate of the Square

e) public int getPosY() Accessor(selector)

Gets the Y coordinate of the Square

f) public void setEmpty(boolean empty) Transformer (mutative)

Sets the value whether the Square is empty or not

g) public boolean getEmpty() Accessor(observer)

Gets the value if a square is empty or not

h) public void setPawn(Pawn p) Transformer: (mutative)

Set the pawn on the square

I) public Pawn getPawn() Accessor(selector):

Gets the pawn that is on the square

Subs classes of Square

class StartSquare :

Αυτή η κλάση αναπαριστά το Start κουτάκι του ταμπλό και κληρονομεί τις μεθόδους και τα attributes της Square.

Επιπλέον attributes και method του StartSquare:

Attributes:

a) private String canEnter; (A String that stores the color of the player that can enter the square).

Methods:

a)public StartSquare(int posX, int posY, String color) Constructor

Creates the StartSquare(calls super contructor and sets the canEnter to the color of the player that can enter it).

b) public String getEnter() Accessor(selector)

Returns the color of the player that can enter the square.

Class HomeSquare :

Αυτή η κλάση αναπαριστά το Home κουτάκι του ταμπλό και κληρονομεί τις μεθόδους και attributes της Square

Επιπλέον attributes και methods της HomeSquare:

Attributes:

a) private String canEnter; (A String that stores the color of the player that can enter the square).

Methods:

a)public HomeSquare(int posX, int posY, String color) Constructor

Creates the HomeSquare(calls super contructor and sets the canEnter to the color of the player that can enter it).

b) public String getEnter() Accessor(selector)

Gets who can enter the square

Class SafetySquare :

Αυτή η κλάση αναπαριστά το Safety κουτάκι του ταμπλό και κληρονομεί τις μεθόδους και attributes της Square

Επιπλέον attributes και methods της SafetySquare:

Attributes:

a) private String canEnter; (A String that stores the color of the player that can enter the square).

Methods:

a)public SafetySquare(int posX, int posY, String color) Constructor

Creates the SafetySquare(calls super constructor and sets the canEnter to the color of the player that can enter it).

b) public String getEnter() Accessor(selector)

Gets who can enter the square

Class SimpleSquare:

Αυτή η κλάση αναπαριστά τα απλά κουτάκια του ταμπλό και κληρονομεί τις μεθόδους και attributes της Square

Αυτή η κλάση δεν έχει κάποια επιπλέον ιδιότητα. Ο constructor καλεί τον super constructor και δίνει το χρώμα άσπρο μαζί με τις συντεταγμένες.

Contructor : public SimpleSquare(int posX, int posY)

Creates a SimpleSquare using super.

Class SlideSquare:

Η κλάση SlideSquare κληρονομεί τα attributes και methods της Square. Δεν έχει κάποια επιπλέον ιδιότητα καθώς ο λόγος ύπαρξης της είναι για να την κάνουν extend οι κλάσεις StartSlideSquare , InternalSlideSquare , EndSlideSquare . Έχει μόνο έναν constructor που χρησιμοποιεί super.

Contrctuctor: public SlideSquare(int posX , int posY, String color)

Creates a SlideSquare using super.

Class StartSlideSquare:

Η κλάση StartSlideSquare κληρονομεί τα attributes και methods της Square.

Επιπλέον attributes και methods:

Attributes:

a) private int length (Stores how many squares to reach the EndSlideSquare).

Methods:

a) public StartSlideSquare(int posX,int posY,String color,int length)

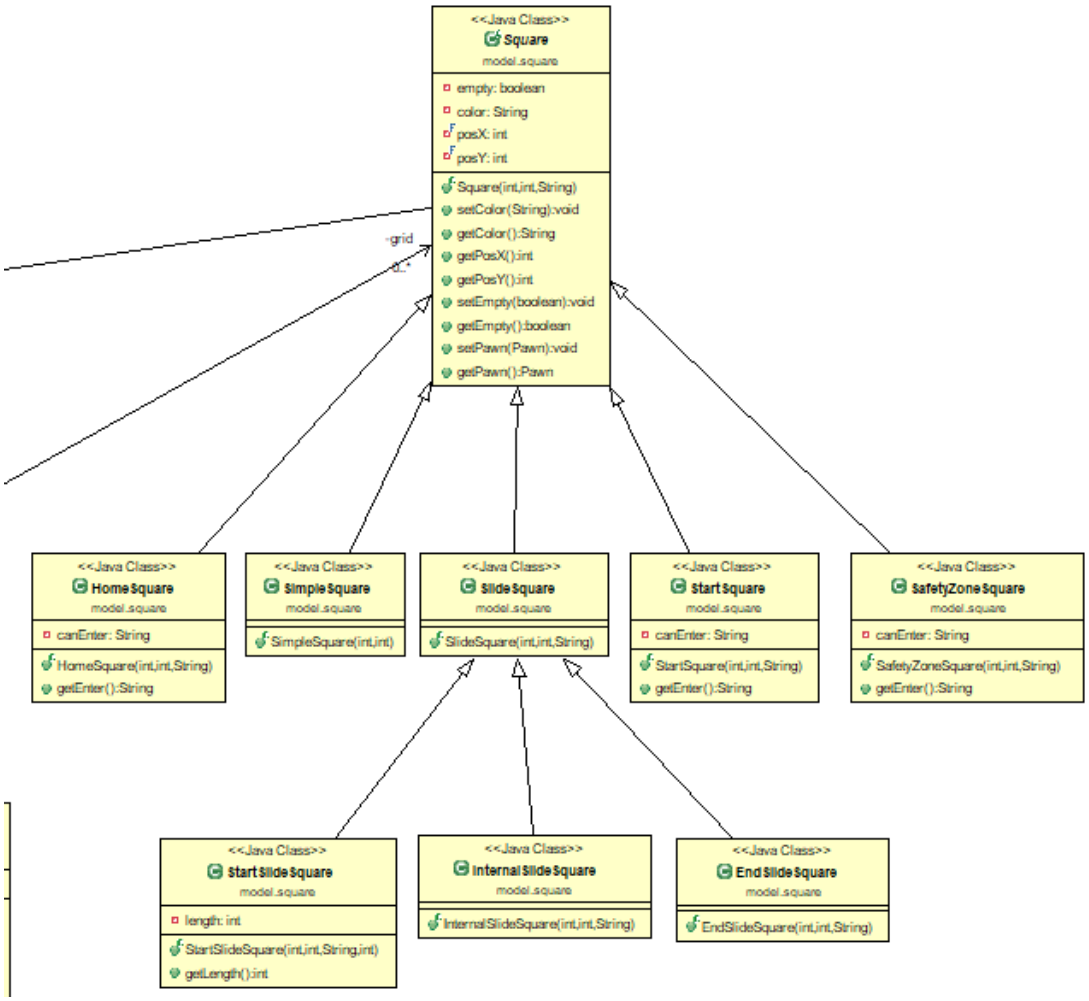
Creates the StartSlideSquare and sets values to the attributes.

b) public int getLength() Accessor(selector)

Returns the length of the slide

Οι κλάσεις InternalSlideSquare και EndSlideSquare δεν έχουν κάποια επιπλέον ιδιότητα και έχουν όλα τα attributes και methods που κληρονομούν

UML διάγραμμα για το Square και τις sub classes του.



Class Player

Αυτή η κλάση αναπαριστά έναν παίχτη του παιχνιδιού.

Attributes :

α) private String color (The color of the player).

b) private Pawn pawn1 ,pawn2(The pawns the player has).

Methods:

a) public Player(int x, int y ,String color) Constructor

Initializes the color and the pawns of the player.

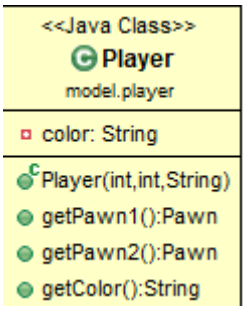
b) public Pawn getPawn1() Accessor(selector)

Returns the pawn1 of the player.

c) public Pawn getPawn2() Accessor(selector)

Returns the pawn2 of the player.

UML για την κλάση Player



Class Pawn

Η κλάση Pawn αναπαριστά ένα πιόνι του παιχνιδιού.

Attributes :

a) private int posX , posY (Coordinates of the current position of the pawn).

b) private int startX ,startY (Coordinates of the starting position of the pawn).

c) private String color (The color of the pawn(basically which player has it)).

d) private boolean active (If the pawn is in the game or not).

e) private int square (The square identifier of the pawn)

Methods:

a) public Pawn(int x , int y, String color) contructor

Creates a pawn and sets the initial values

b) public void move(int x, int y)

Moves the pawn to the square with x,y coordinates.

c) public void setPosX(int x) Transformer(mutative)

Sets the X coordinate for the pawn.

d) public void setPosY(int y) Transformer(mutative)

Sets the Y coordinate for the pawn.

e)public void setColor(String c) Transformer(mutative)

Sets the color for the pawn.

f) public void setActive(boolean active) Transformer(mutative)

Sets the activity of the pawn.

g) public int getPosX Accessor(selector)

Returns the X coordinate of the pawn.

h) public int getPosY Accessor(selector)

Returns the Y coordinate of the pawn.

i) public String getColor Accessor(selector)

Returns the color of the pawn.

j) public int getStartX Accessor(selector)

Returns the starting X coordinate of the pawn.

k) public int getStartY Accessor(selector)

Returns the starting Y coordinate of the pawn.

l) public boolean getActive Accessor(observer)

Returns the activity of the pawn.

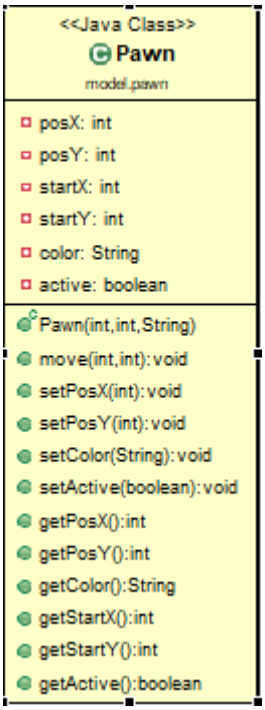
m) public void setSquare(int s) Transformer(mutative)

Sets the square the pawn sits on

n) public int getSquare() Accessor(observer)

Gets the square that the pawn sits on

UML για την κλάση Pawn:



Class Deck

Η κλάση Deck περιέχει το ταμπλό του παιχνιδιού(Pawns , Players, Cards).

Attributes :

a) private Square[] grid;(Contains all the squares of the game).

b) private Player redPlayer,yellowPlayer; (The two players of the game).

c) private Stack<Card> stackOfCards; (All of the cards in the game).

d) private Card currentCard; (The card the player just drew).

e) private String choice; (The choice of the player used in cards)

f) private Player whoPlays; (Which player is playing right now)

g) private int slide ; (Used to store the value if a pawn used a slide or not)

Methods:

a) public boolean CheckIfStackEmpty(Stack<Card> stackOfCards)
Accessor(observer).

Returns true if stack is empty else returns false.

b) public Stack<Card> fillStack() Transformer(mutative)

Fills the stack with cards and then returns it

c) public Stack<Card> Shuffle() Transformer(mutative)

Shuffles the cards and returns the stack.

d) public Player getRedPlayer() Accessor(selector)

Returns the red player

e) public Player getYellowPlayer() Accessor(selector)

Returns the yellow player

f) public Stack<Card> getStack() Accessor(selector)

Returns the stack of cards

g) public int getStackSize() Accessor(selector)

Gets how many cards left in the stack

h) public Card peekTopCard() Accessor(selector)

peeks the top card

i) public Card getCurrentCard() Accessor(selector)

gets the current card

j) public void setCurrentCard(Card c) Transformer(mutative)

Sets the current card

k) public Square getGrid(int I) Accessor(selector)

gets the grid[i]

l) public boolean doSlide(Pawn p ,Deck b,int I)

Check is the pawn lands on a StartSlideSquare if it lands on StartSlideSquare then moves to EndSlideSquare else it stays on the square. Also if the StartSlideSquare is of same color then it stays on the square.

m) public void setOption(String choice) Transformer(mutative)

Sets the option of the player

n) public String getOption() Accesor(selective)

gets the option of the player

o) public void setWhoPlays(Player currentPlayer) Transformer(mutative)

Sets who is playing right now

p) public Player getWhoPlays() Accessor(selector)

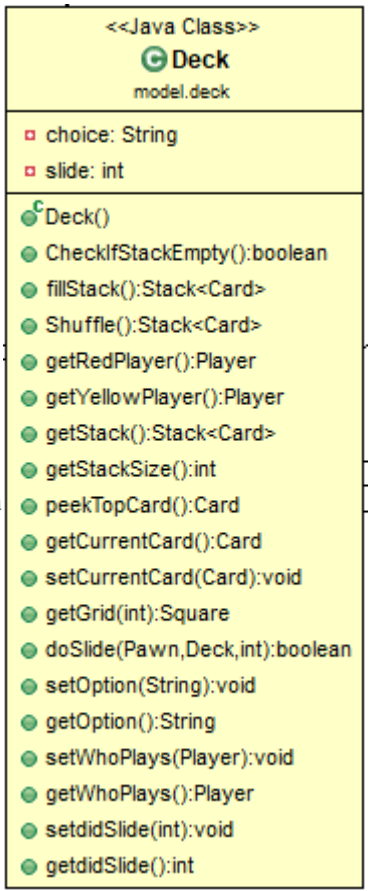
Gets who is playing right now

q) public void setdidSlide(int slide) Transformer(mutative)

Sets the value slide

r) public int getdidSlide() Accessor(selector)

Method to get if a pawn did a slide or not(0 if no pawns slided, 1 if one pawn slided and 2 if 2 pawns slided)



UML of the class Deck.

3. Η Σχεδίαση και οι Κλάσεις του Πακέτου Controller

Εδώ θα περιγράψετε το σχέδιο υλοποίησης της προγραμματιστικής εργασίας του πακέτου controller. Συγκεκριμένα, στην Α φάση του Project θα περιγράψετε ποιες μεθόδους σκοπεύετε να χρησιμοποιήσετε στη Β φάση του Project (και για ποιο λόγο είναι χρήσιμες, ποια είναι η λειτουργικότητα τους), ενώ θα πρέπει να εξηγήσετε με ποιο τρόπο ο controller θα αλληλεπιδρά με το model και το view. Αντίστοιχα στη Β φάση του project, συμπληρώνετε τις λεπτομέρειες της υλοποίησης και τυχόν αλλαγές.

Class Controller

Η κλάση Controller λειτουργεί σαν τον εγκέφαλο του παιχνιδιού. Είναι υπεύθυνη για να ξεκινήσει το παιχνίδι και να δημιουργήσει όλα τα απαραίτητα στιγμιότυπα. Μέσω του View παίρνει τις επιλογές των παιχτών και κάνει τις κατάλληλες ενέργειες. Επίσης είναι υπεύθυνη για να ανακατεύει την τράπουλα όταν αυτή τελειώνει και επίσης θα κηρύξει τον νικητή όταν υπάρξει.

Attributes :

a) private Pawn[] redPawns,yellowPawns(The pawns of each player).

b) private Card currentCard(The card the player just drew).

c) private Pawn pickedPawn(The pawn the player just clicked).

d) private Player whoPlays(The player that is playing right now).

e) private Player winnerPlayer (The winner of the game).

f) private String winner (The color of the winner);

g) private String playerChoice;(The choice the player made on card movements)

h) private Deck deck;(instance of the Deck class)

Methods:

a) public void drawCard() Transformer(mutative)

This method pops a card from the stack

b) public void checkWinner() Transformer(mutative).

Checks if there is a winner in the game.

c) public String getWinner() Accessor (selector)

Gets the winner of the game(the color)

d) public Player getWinnerPlayer() Accessor(selector)

Gets the winner of the game in Player class form

e) public void setWhoPlays(Player currentPlayer) Transfomer(mutative)

Sets the whoPlays(Player that is playing right now).

f) public Player getWhoPlays() Accessor(selector)

Gets who is playing right now.

g) public Deck getDeck() Accessor(selector)

Gets the Deck instance

h) public void setPickedPawn(int pawn) Transformer(mutative)

Sets the pawn that was picked

I) public Pawn getPickedPawn() Accessor(selector)

Gets the picked pawn

j) public Card getCurrentCard() Accessor(selector)

Gets the current card

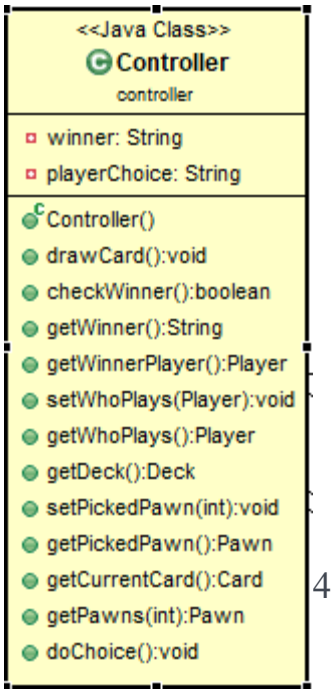
k) public Pawn getPawns(int pawn) Accessor(selector)

Gets a pawn depending on a the parameter

l) public void doChoice()

Gets and stores the option of the player

UML controller :



4. Η Σχεδίαση και οι Κλάσεις του Πακέτου View

Class GUI

Attributes:

private Controller	game; Instance of Controler
private myDesktopPane	panel; panel that has all GUI components
private URL	imageURL, soundURL; paths to lead sound and images
private JButton	FoldButton; Button for Fold
private JButton	drawcard; Button to draw card
private JLabel	text1,text2; texts
private ClassLoader	cldr; instance of class loader
private JDesktopPane	blueBack; blue background inside the board
private JTextArea	textArea,winnerLabel; text area and winner test area
private JButton[]	pawns; buttons of pawns
private JLabel	sorryImage,musicIcon; images and music icon
private JLabel[]	squares; the squares of the game

private JLabel yellowHome , redHome Home locations

private JLabel yellowStart , redStart; Start locations

private int counterX = 0 , counterY = 0;

private String turn,cardsLeft,description;

private JLabel currentCard;

private JLabel message10;

private int card10 = 0;

private JFrame parent;

private JSlider volumeSlider;

private Clip clipMusic;

Methods :

a) private void initComponents() transformer(mutative)

initializes some buttons and labels

b) public void playSound(String sound,double volume)

Method to play sounds of the game

c) private static void setVol(double volume,Clip clip) Transfomer(mutative)

Change the volume of the game

Action Listeners:

private class CardListener implements ActionListener

Does the actions required when the card button is pressed

private class FoldListener implements ActionListener

Does the actions required to fold

private class PawnListener implements ActionListener

Does the actions required when a pawn is clicked

private class VolumeListener implements ChangeListener

Does the actions required to adjust the volume

Class OptionDialog

Attributes:

a) private String s; String that stores the decision of the Player

b) private URL imageURL URL to load an image

c) ClassLoader cldr; class loader to load an image

Methods:

a) public OptionDialog(int value) Constructor

Creates a new Option Dialog Window for cards 7 AND 10

b) public OptionDialog(int value ,boolean redPawn1,boolean redPawn2,

boolean yellowPawn1 , boolean yellowPawn2)

Creates a new Option Dialog Window for card 11

c) public OptionDialog(int value ,boolean redPawn1,boolean redPawn2,

boolean yellowPawn1 , boolean yellowPawn2 ,String color)

Creates a new Option Dialog Window for card Sorry

d) public String choice() accessor(selector)

Returns the choice of a player

5. Η Αλληλεπίδραση μεταξύ των κλάσεων – Διαγράμματα UML

Σε αυτήν την ενότητα μπορείτε να συμπεριλάβετε διαγράμματα UML, και να εξηγήσετε μέσω αυτών την αλληλεπίδραση των κλάσεων (πχ μεταξύ των κλάσεων διαφορετικών πακέτων.

Το διάγραμμα UML ειναι στον φάκελο της άσκησης(Δεν μπορώ να το βάλω μέσα σε αυτό το docx γιατί είναι μεγάλο).Το Controller αλληλεπιδρά με όλα τα πακέτα αφού είναι υπεύθυνο για να γίνεται σωστά η σύνδεση με τον View και Model.

6. Λειτουργικότητα (Β Φάση)

Σε αυτήν την ενότητα θα γράψετε στη Β φάση ποια ερωτήματα καταφέρατε να υλοποιήσετε είτε επιτυχώς είτε εν μέρει (και ενδεχομένως ποια όχι).

Υλοποίησα όλες τις λειτουργίες του προγράμματος. Δεν έκανα Junit tests. Το παιχνίδι ξεκινάει όταν πατιέται το κουμπί για να τραβηχτεί η πρώτη κάρτα. Ο κάθε παίχτης πρέπει να δοκιμάσει να κάνει κίνηση και με τα 2 πιόνια και αν δεν έχει νόμιμη κίνηση τότε θα αλλάξει η σειρά αυτόματα στον επόμενο παίχτη. Οι κάρτες είναι φτιαγμένες έτσι ώστε όταν αλλάζει η σειρά να τραβιούνται αυτόματα χωρίς να χρειάζεται να κάνει κλικ ο χρήστης(εκτός της πρώτης φοράς). Επίσης για να πάρει το παιχνίδι την απόφαση του κάθε παίχτη για κάρτες που δεν έχουν αποκλειστική κίνηση(κάρτες 7, 10, 11, Sorry) το παιχνίδι εμφανίζει ένα παράθυρο διαλόγου.

7. Συμπεράσματα

Γενικά είχα πρόβλημα να καταλάβω από την εκφώνηση κάποια πράγματα που πρότεινε ο καθηγητής στην εκφώνηση(Οπως το να έχουμε μέθοδο να κουνάει πιόνια και στις κάρτες(movePawn) , και στο Deck και στο Controller).Επίσης είχα θέμα για να φτιάξω το UML επειδή το ObjectAid είναι προβληματικό και το licence δεν έκανε renew για να το χρησιμποιήσω