Πανεπιστήμιο Κρήτης – Τμήμα Επιστήμης Υπολογιστών

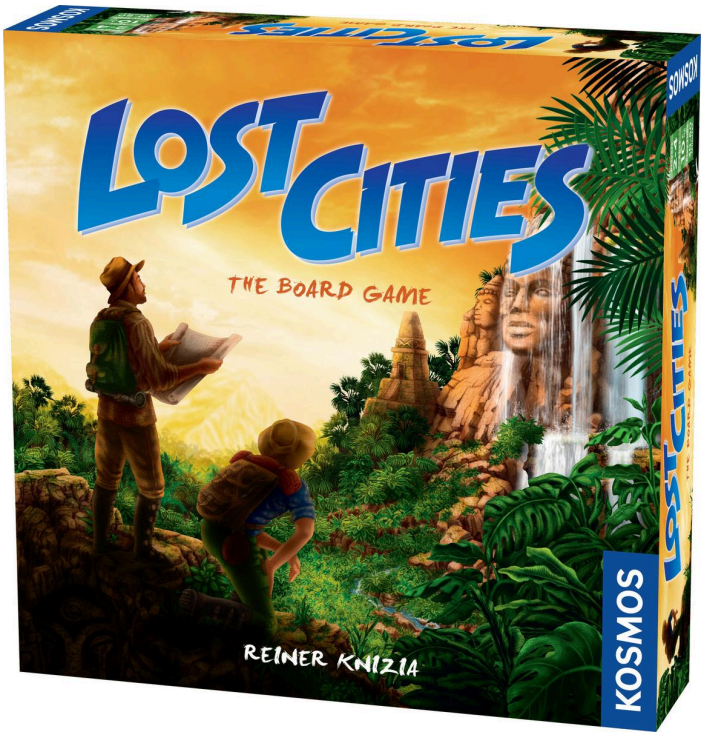
Αντικειμενοστραφής Προγραμματισμός

Διδάσκων: Ι. Τζίτζικας

Χειμερινό Εξάμηνο 2024-2025

Α’ ΦΑΣΗ PROJECT

Υλοποίηση επιτραπέζιου παιχνιδιού “Lost Cities”



Παπαδάκης Γεώργιος 4975
Δεκέμβριος 2024

Περιεχόμενα

1. Εισαγωγή

Σε αυτό το project υλοποιούμε το επιτραπέζιο παιχνίδι "Lost Cities" το οποίο στα πλαίσια του μαθήματος έχει θέμα τον Μινωικό Πολιτισμό.

Για τη δημιουργία του παιχνιδιού χρησιμοποιούμε το προγραμματιστικό μοντέλο MVC (Model View Controller), βάσει του οποίου η υλοποίηση χωρίζεται σε 3 ενότητες:

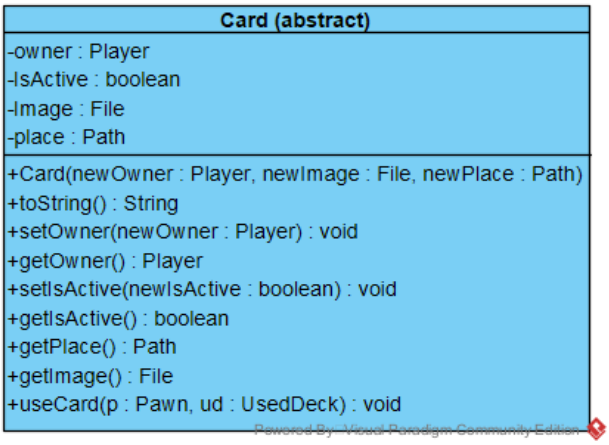
1. Model: Η υλοποίηση της λειτουργικότητας του παιχνιδιού. Δηλαδή το πως λειτουργούν οι κάρτες, τα πιόνια, αλλαγές στο ταμπλό, κ.α.
2. View: Η υλοποίηση της εμφάνισης στον χρήστη. Δηλαδή τα UI και GUI.
3. Controller: Ο κώδικας που διαχειρίζεται αποτελεσματικά τα κομμάτια των Model και VIew, καθώς και την μεταξύ τους επικοινωνία, ώστε να παίζουμε το παιχνίδι κανονικά.

Σε αυτή την αναφορά θα παρουσιαστεί ο τρόπος υλοποίησης κάθε ενός από τα κομμάτια του μοντέλου MVC, μαζί με διαγράμματα UML και κομμάτια κώδικα, συνοδευμένα από βοηθητικά σχόλια τύπου Javadoc.

2. Η Σχεδίαση και οι Κλάσεις του Πακέτου Model

Το model θα πρέπει να περιλαμβάνει όλα τα περιεχόμενα του παιχνιδιού, δηλαδή θα πρέπει να υπάρχουν κλάσεις για τις κάρτες, τις θέσεις του ταμπλό του παιχνιδιού, τον παίκτη, τα πιόνια του παίκτη, το ταμπλό κλπ. Ας δούμε την κάθε κλάση αναλυτικά:

- **Card**



Η αφηρημένη κλάση Card αποτελεί τη βάση για κάθε κάρτα του παιχνιδιού.

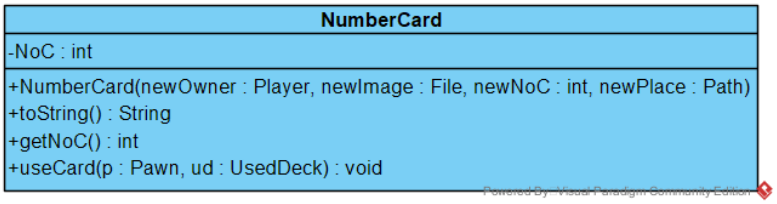
Δεδομένα:

- *"private Player Owner"* Ο ιδιοκτήτης της κάρτας. Μπορεί να είναι τύπου Player ή null αν δεν έχει χρησιμοποιηθεί ακόμα.
- *"private boolean IsActive"* Αν η κάρτα έχει παιχτεί στο παρελθόν τότε η isActive είναι FALSE, αλλιώς είναι TRUE.
- *"private File Image"* Η εικόνα που αναπαριστά την εκάστοτε κάρτα.
- *"private Path place"* Το μονοπάτι στο οποίο ανήκει αυτή η κάρτα.

Μέθοδοι:

- *"public Card(Player newOwner, File newImage, Path newPlace)"* Ο constructor της κλάσης.
- *"public String toString()"* Πληροφορίες σχετικά με τη κάρτα.
- *"public void setOwner(Player newOwner)"* Θέτει καινούργιο ιδιοκτήτη για τη κάρτα.
- *"public Player getOwner()"* Επιστρέφει τον ιδιοκτήτη της κάρτας.
- *"public void setIsActive(boolean newIsActive)"* Αλλάζει την διαθεσιμότητα της κάρτας. Αν έχει παιχτεί τότε η IsActive γίνεται FALSE.
- *"public boolean getIsActive()"* Επιστρέφει boolean σχετικά με το αν η κάρτα είναι ενεργή ή όχι.
- "*public Path getPlace()*" Επιστρέφει το μονοπάτι στο οποίο ανήκει η κάρτα.
- "*public File getImage()*" Επιστρέφει την εικόνα της κάρτας.
- "*public void useCard(Pawn p, UsedDeck ud)*" Ελέγχει αν η κάρτα μπορεί να παιχτεί, μετακινεί το πιόνι στη σωστή θέση και εναποθέτει τη κάρτα στη στοίβα με τις χρησιμοποιημένες κάρτες.

- **NumberCard**



Η κλάση NumberCard αφορά τις απλές, αριθμημένες κάρτες του παιχνιδιού.

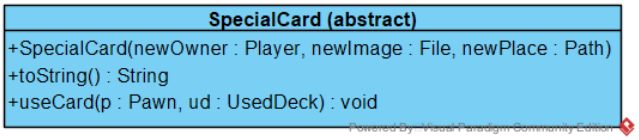
Δεδομένα:

- *"private int NoC"* Ο αριθμός της κάρτας.

Μέθοδοι:

- *"public NumberCard(Player newOwner, File newImage, int newNoC, Path newPlace)"* Ο constructor της κλάσης.
- *"public String toString()"* Πληροφορίες σχετικά με τη κάρτα.
- *"public int getNoC()"* Επιστρέφει τον αριθμό της κάρτας.
- "*public void useCard(Pawn p, UsedDeck ud)*" Ελέγχει αν η κάρτα μπορεί να παιχτεί, μετακινεί το πιόνι στη σωστή θέση και εναποθέτει τη κάρτα στη στοίβα με τις χρησιμοποιημένες κάρτες.

- **SpecialCard**

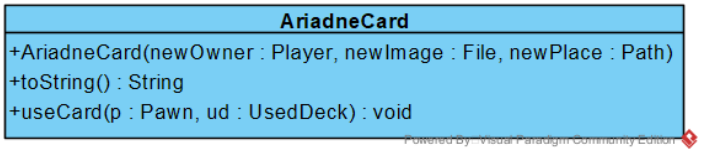


Η αφηρημένη κλάση SpecialCard αφορά τις κάρτες επιπλέον των αριθμημένων, δηλαδή τη κάρτα-Αριάδνη και τη κάρτα-Μινώταυρο.

Μέθοδοι:

- *"public SpecialCard(Player newOwner, File newImage, Path newPlace)"* Ο constructor της κλάσης.
- *"public String toString()"* Πληροφορίες σχετικά με τη κάρτα.
- "*public void useCard(Pawn p, UsedDeck ud)*" Ελέγχει αν η κάρτα μπορεί να παιχτεί, μετακινεί το πιόνι στη σωστή θέση και εναποθέτει τη κάρτα στη στοίβα με τις χρησιμοποιημένες κάρτες.

- **AriadneCard**

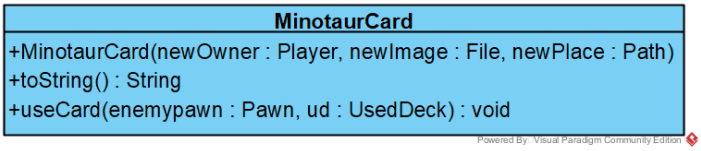


Η κλάση AriadneCard αφορά τη κάρτα-Αριάδνη.

Μέθοδοι:

- *"public AriadneCard(Player newOwner, File newImage, Path newPlace)"* Ο constructor της κλάσης.
- *"public String toString()"* Πληροφορίες σχετικά με τη κάρτα.
- "*public void useCard(Pawn p, UsedDeck ud)*" Ελέγχει αν η κάρτα μπορεί να παιχτεί, μετακινεί το πιόνι στη σωστή θέση και εναποθέτει τη κάρτα στη στοίβα με τις χρησιμοποιημένες κάρτες.

❖ MinotaurCard

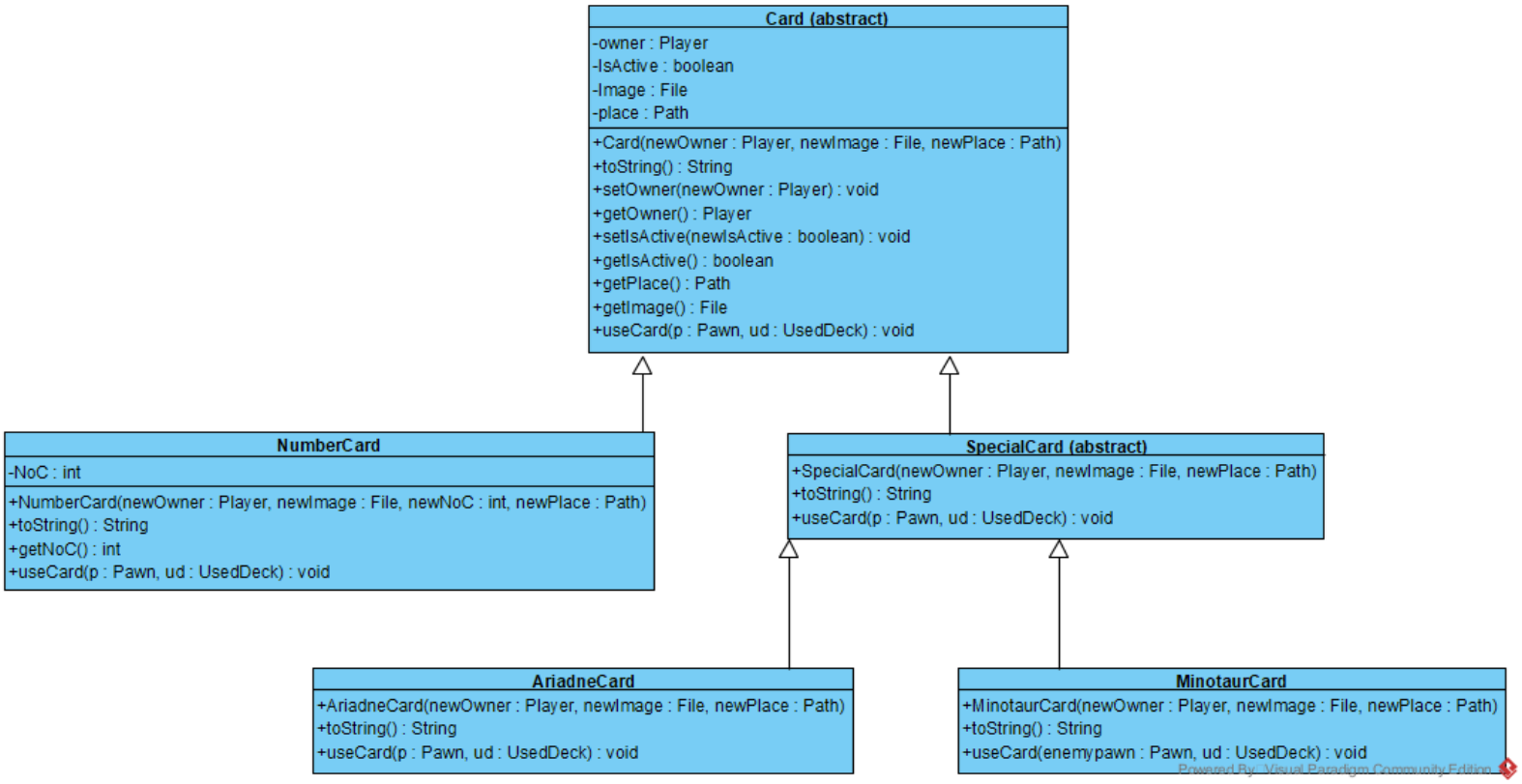


Η κλάση MinotaurCard αφορά τη κάρτα-Μινώταυρο.

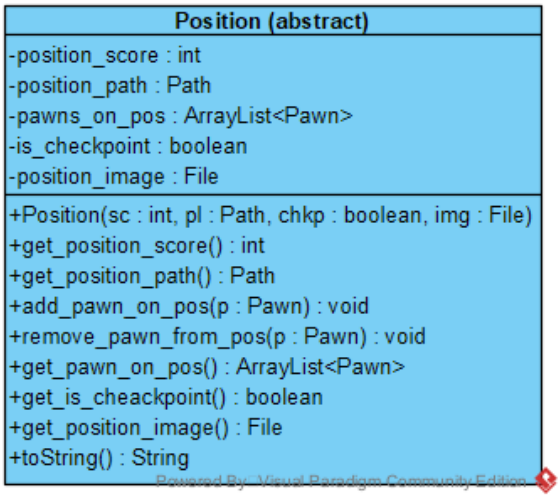
Μέθοδοι:

- *"public MinotaurCard(Player newOwner, File newImage, Path newPlace)"* Ο constructor της κλάσης.
- *"public String toString()"* Πληροφορίες σχετικά με τη κάρτα.
- "*public void useCard(Pawn p, UsedDeck ud)*" Ελέγχει αν η κάρτα μπορεί να παιχτεί, μετακινεί το πιόνι στη σωστή θέση και εναποθέτει τη κάρτα στη στοίβα με τις χρησιμοποιημένες κάρτες.

ΠΛΗΡΕΣ ΔΙΑΓΡΑΜΜΑ ΤΩΝ ΚΛΑΣΕΩΝ (ΜΕ ΤΗ ΚΛΗΡΟΝΟΜΙΚΟΤΗΤΑ ΤΟΥΣ)



- **Position**



Η αφηρημένη κλάση Position αποτελεί τη βάση για κάθε θέση του παιχνιδιού.

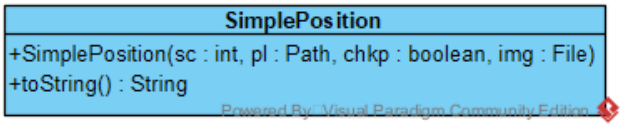
Δεδομένα:

- *"private int position_score"* Το score της κάθε θέσης.
- *"private Path position_path"* Το μονοπάτι στο οποίο ανήκει η κάθε θέση.
- *"private ArrayList<Pawn> pawns_on_pos"* Λίστα με όλα τα πιόνια που είναι πάνω στη θέση.
- *"private boolean is_checkpoint"* Boolean η οποία έχει τιμή TRUE εάν η θέση είναι στο checkpoint ή μετά από αυτό, και τη τιμή FALSE αν είναι πριν το checkpoint.
- *"private File position_image"* Η εικόνα της θέσης.

Μέθοδοι:

- *"public Position(int sc, Path pl, boolean chkp, File img)"* Ο constructor της κλάσης.
- *"public int get_position_score()"* Επιστρέφει το score της θέσης.
- *"public Path get_position_path()"* Επιστρέφει το μονοπάτι της θέσης.
- *"public void add_pawn_on_pos(Pawn p)"* Τοποθετεί το πιόνι p στη λίστα με τα πιόνια που φιλοξενεί αυτή η θέση.
- *"public void remove_pawn_from_pos(Pawn p)"* Αφαιρεί το πιόνι p από τη λίστα με τα πιόνια που φιλοξενεί αυτή η θέση.
- *"public ArrayList<Pawn> get_pawn_on_pos()"* Επιστρέφει τη λίστα με τα πιόνια που φιλοξενεί αυτή η θέση.
- "*public boolean get_is_checkpoint()*" Επιστρέφει τη boolean τιμή για το εάν η θέση αυτή είναι checkpoint ή όχι.
- "*public File get_position_image()*" Επιστρέφει την εικόνα της θέσης.
- "*public String toString()"* Πληροφορίες σχετικά με τη θέση.

- **SimplePosition**

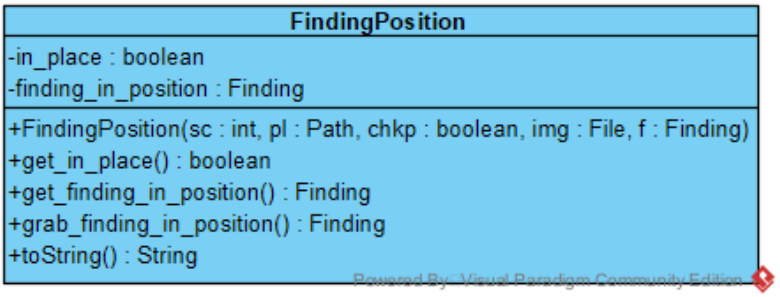


Η κλάση SimplePosition αφορά τις θέσεις οι οποίες **δε** κρύβουν κάποια αρχαιότητα.

Μέθοδοι:

- *"public SimplePosition(int sc, Path pl, boolean chkp, File img)"* Ο constructor της κλάσης.
- *"public String toString()"* Πληροφορίες σχετικά με τη θέση.

❖ FindingPosition



Η κλάση FindingPosition αφορά τις θέσεις οι οποίες κρύβουν κάποια αρχαιότητα.

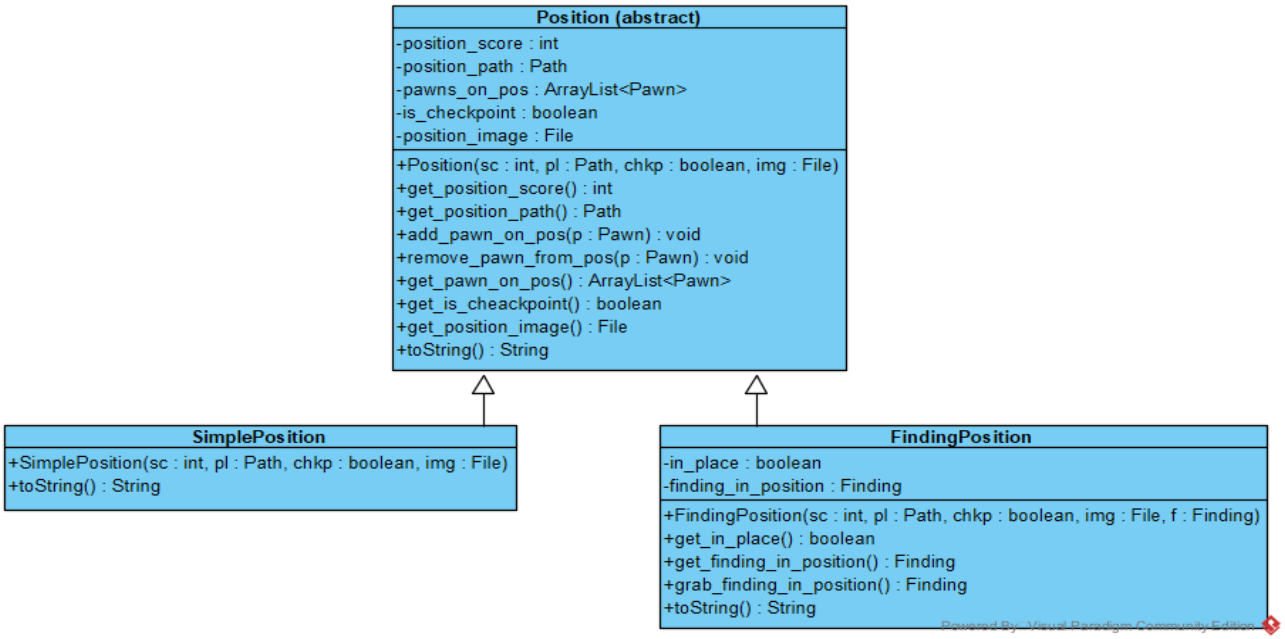
Δεδομένα:

- *"private boolean in_place"* Boolean η οποία έχει τιμή TRUE αν η αρχαιότητα είναι ακόμα κρυμμένη σε αυτή τη θέση, και τιμή FALSE αν κάποιος παίκτης την έχει πάρει ή την έχει καταστρέψει.
- *"private Finding finding_in_position"* Η αρχαιότητα η οποία είναι κρυμμένη σε αυτή τη θέση.

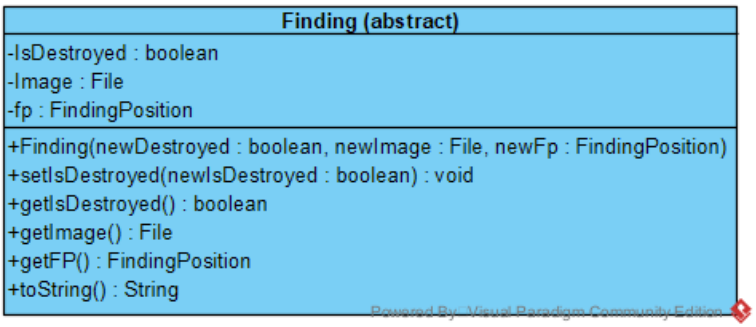
Μέθοδοι:

- *"public FindingPosition(int sc, Path pl, boolean chkp, File img, Finding f)"* Ο constructor της κλάσης.
- *"public boolean get_in_place()"* Επιστρέφει τη διαθεσιμότητα της αρχαιότητας αυτής της θέσης.
- *"public Finding get_finding_in_position()"* Επιστρέφει την αρχαιότητα που κρύβεται σε αυτή τη θέση(χωρίς να την αφαιρεί).
- *"public Finding grab_finding_in_position()"* Επιστρέφει την αρχαιότητα που κρύβεται σε αυτή τη θέση, βάζει στη μεταβλητή finding_in_position τιμή null και στη μεταβλητή in_place τιμή FALSE. (χρησιμοποιείται όταν κάποιος παίκτης μπορεί να πάρει την αρχαιότητα και δεν είναι τοιχογραφία)
- "*public String toString()"* Πληροφορίες σχετικά με τη θέση.

ΠΛΗΡΕΣ ΔΙΑΓΡΑΜΜΑ ΤΩΝ ΚΛΑΣΕΩΝ (ΜΕ ΤΗ ΚΛΗΡΟΝΟΜΙΚΟΤΗΤΑ ΤΟΥΣ)



- **Finding**



Η αφηρημένη κλάση Finding αποτελεί τη βάση για κάθε αρχαιότητα του παιχνιδιού.

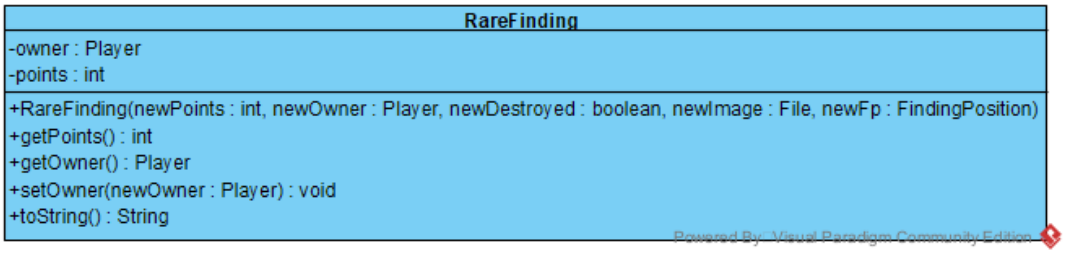
Δεδομένα:

- *"private boolean IsDestroyed"* Boolean η οποία έχει τιμή TRUE εάν η αρχαιότητα έχει καταστραφεί από τον Θησέα, και τη τιμή FALSE αν έκατσε ήρεμα σήμερα ο αθηναίος.
- *"private File Image"* Η εικόνα της αρχαιότητας.
- *"private FindingPosition fp"* Η θέση στην οποία κρύβεται αυτή η αρχαιότητα.

Μέθοδοι:

- *"public Finding(boolean newDestroyed, File newImage, FindingPosition newFp)"* Ο constructor της κλάσης.
- *"public void setIsDestroyed(boolean newIsDestroyed)"* Θέτει καινούργια τιμή στη boolean IsDestroyed.
- *"public boolean getIsDestroyed()"* Επιστρέφει τη τιμή της boolean IsDestroyed.
- *"public File getImage()"* Επιστρέφει την εικόνα της αρχαιότητας.
- *"public FindingPosition getFP()"* Επιστρέφει τη θέση στην οποία κρύβεται η αρχαιότητα αυτή.
- "*public String toString()"* Πληροφορίες σχετικά με την αρχαιότητα.

❖ **RareFinding**



Η κλάση RareFinding αφορά τις 4 πολύτιμες αρχαιότητες, καθεμία από τις οποίες κρύβεται και σε ένα από τα 4 μονοπάτια του παιχνιδιού.

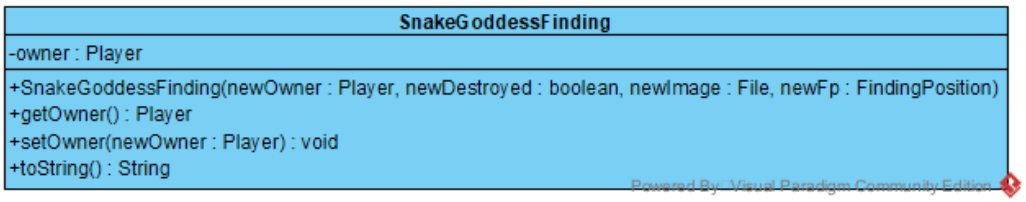
Δεδομένα:

- *"private Player owner"* Ο ιδιοκτήτης αυτής της αρχαιότητας. Αν δεν έχει ανακαλυφθεί ακόμα έχει τιμή null.
- *"private int points"* Οι βαθμοί που κερδίζει όποιος ανακαλύψει αυτή την αρχαιότητα.

Μέθοδοι:

- *"public RareFinding(int newPoints, Player newOwner, boolean newDestroyed, File newImage, FindingPosition newFp)"* Ο constructor της κλάσης.
- *"public int getPoints()"* Επιστρέφει τον αριθμό των πόντων που κερδίζει όποιος παίκτης ανακαλύψει αυτή την αρχαιότητα.
- *"public Player getOwner()"* Επιστρέφει τον ιδιοκτήτη αυτής της αρχαιότητας.
- *"public void setOwner(Player newOwner)"* Θέτει ιδιοκτήτη για αυτή την αρχαιότητα.
- "*public String toString()"* Πληροφορίες σχετικά με την αρχαιότητα.

❖ **SnakeGoddessFinding**



Η κλάση SnakeGoddessFinding αφορά τα 10 αγαλματάκια της Θεάς των Φιδιών τα οποία είναι κρυμμένα σε 10 θέσεις στα μονοπάτια του παιχνιδιού.

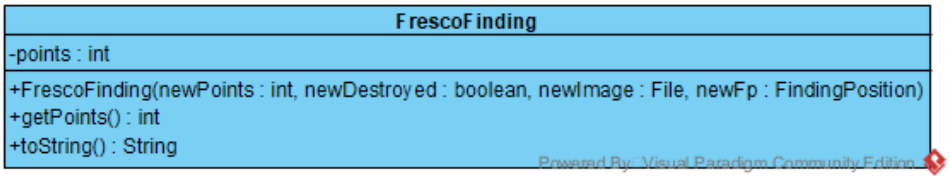
Δεδομένα:

- *"private Player owner"* Ο ιδιοκτήτης αυτής της αρχαιότητας. Αν δεν έχει ανακαλυφθεί ακόμα έχει τιμή null.

Μέθοδοι:

- *"public SnakeGoddessFinding(Player newOwner, boolean newDestroyed, File newImage, FindingPosition newFp)"* Ο constructor της κλάσης.
- *"public Player getOwner()"* Επιστρέφει τον ιδιοκτήτη αυτής της αρχαιότητας.
- *"public void setOwner(Player newOwner)"* Θέτει ιδιοκτήτη για αυτή την αρχαιότητα.
- "*public String toString()"* Πληροφορίες σχετικά με την αρχαιότητα.

❖ **FrescoFinding**



Η κλάση FrescoFinding αφορά τις 6 τοιχογραφίες οι οποίες είναι κρυμμένες σε 6 θέσεις στα μονοπάτια του παιχνιδιού.

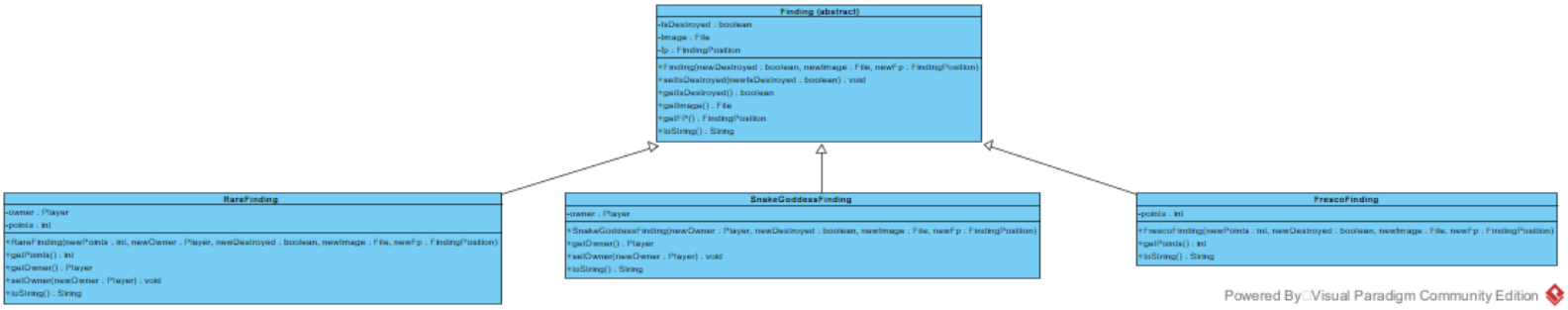
Δεδομένα:

- *"private int points"* Οι βαθμοί που κερδίζει όποιος ανακαλύψει αυτή την αρχαιότητα.

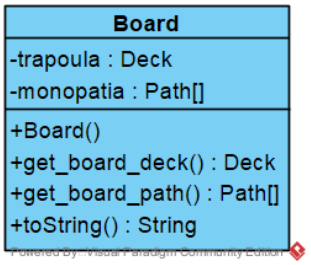
Μέθοδοι:

- *"public FrescoFinding(int newPoints, boolean newDestroyed, File newImage, FindingPosition newFp)"* Ο constructor της κλάσης.
- *"public int getPoints()"* Επιστρέφει τον αριθμό των πόντων που κερδίζει όποιος παίκτης ανακαλύψει αυτή την αρχαιότητα.
- "*public String toString()"* Πληροφορίες σχετικά με την αρχαιότητα.

ΠΛΗΡΕΣ ΔΙΑΓΡΑΜΜΑ ΤΩΝ ΚΛΑΣΕΩΝ (ΜΕ ΤΗ ΚΛΗΡΟΝΟΜΙΚΟΤΗΤΑ ΤΟΥΣ)



❖ **Board**



Η κλάση Board αφορά το ταμπλό του παιχνιδιού.

Δεδομένα:

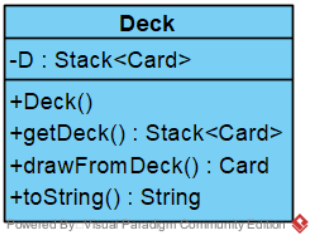
- *"private Deck trapoula"* Η τράπουλα με τις αχρησιμοποίητες κάρτες του παιχνιδιού.
- *"private Path[] monopatia"* Πίνακας με τα 4 μονοπάτια του παιχνιδιού.

Μέθοδοι:

- *"public Board()"* Ο constructor της κλάσης.
- *"public Deck get_board_deck()"* Επιστρέφει τη τράπουλα του παιχνιδιού.

- *"public Path[] get_board_path()"* Επιστρέφει τον πίνακα με τα 4 μονοπάτια του παιχνιδιού.
- "*public String toString()"* Πληροφορίες σχετικά με το ταμπλό.

❖ **Deck**



Η κλάση Deck αφορά τη τράπουλα με τις κάρτες οι οποίες **δεν** έχουν παιχτεί ακόμα.

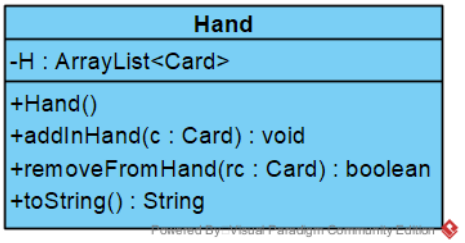
Δεδομένα:

- *"private Stack<Card> D"* Στοίβα η οποία αποθηκεύει τα χαρτιά που **δεν** έχουν χρησιμοποιηθεί ακόμα.

Μέθοδοι:

- *"public Deck()"* Ο constructor της κλάσης.
- *"public Stack<Card> getDeck()"* Επιστρέφει τη στοίβα με τα χαρτιά.
- *"public Card drawFromDeck()"* Δίνει τη κάρτα στη κορυφή της τράπουλας. Κάνει pop από τη στοίβα και δίνει τη κάρτα στον παίκτη που κάλεσε αυτή τη μέθοδο.
- "*public String toString()"* Πληροφορίες σχετικά με τη τράπουλα.

❖ **Hand**



Η κλάση Hand αφορά τις 8 κάρτες που έχει στο χέρι του κάθε παίκτης.

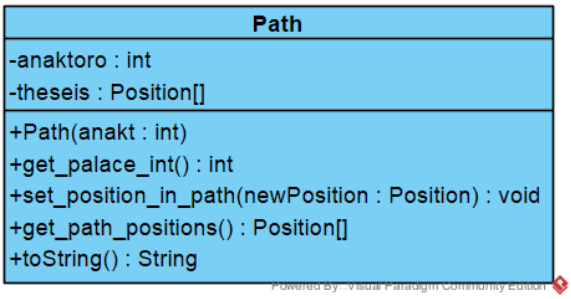
Δεδομένα:

- *"private ArrayList<Card> H"* Λίστα με τα 8 χαρτιά που κρατάει κάθε παίκτης.

Μέθοδοι:

- *"public Hand()"* Ο constructor της κλάσης.
- *"public void addInHand(Card c)"* Τοποθετεί τη κάρτα c στο χέρι του παίκτη.
- *"public boolean removeFromHand(Card rc)"* Αφαιρεί τη κάρτα rc από το χέρι του παίκτη.
- "*public String toString()"* Πληροφορίες σχετικά με το χέρι κάθε παίκτη.

❖ **Path**



Η κλάση Path αφορά κάθε ένα από τα 4 μονοπάτια του παιχνιδιού.

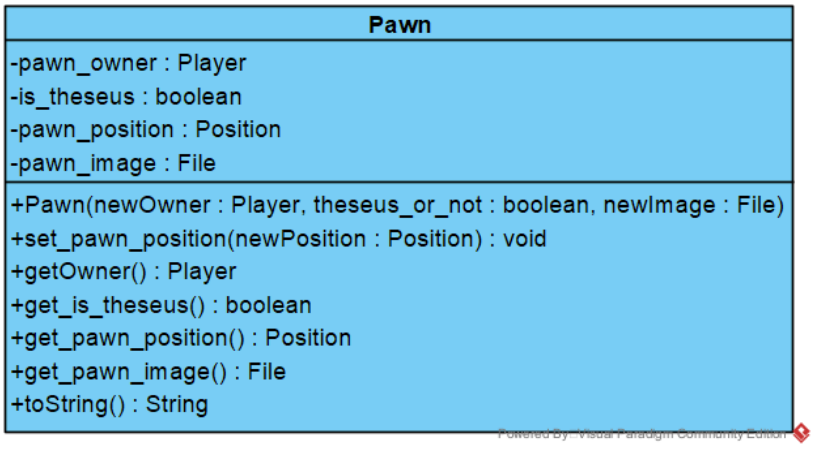
Δεδομένα:

- *"private int anaktoro"* Ακέραια μεταβλητή η οποία υποδεικνύει το ανάκτορο στο οποίο ανήκει αυτό το μονοπάτι. (1=Κνωσός, 2=Μάλια, 3=Φαιστός, 4=Ζάκρος)
- *"private Position[] theseis"* Πίνακας με τις 9 θέσεις κάθε μονοπατιού.

Μέθοδοι:

- *"public Path(int anakt)"* Ο constructor της κλάσης.
- *"public int get_palace_int()"* Επιστρέφει την ακέραια μεταβλητή που υποδεικνύει το ανάκτορο στο οποίο ανήκει το μονοπάτι αυτό.
- *"public void set_position_in_path(Position newPosition)"* Αποθηκεύει τη θέση newPosition στον πίνακα με τις 9 θέσεις του μονοπατιού.
- *"public Position[] get_path_positions()"* Επιστρέφει τον πίνακα με τις θέσεις του μονοπατιού.
- "*public String toString()"* Πληροφορίες σχετικά με το μονοπάτι.

❖ **Pawn**



Η κλάση Pawn αφορά κάθε ένα από τα 8 πιόνια του παιχνιδιού.

Δεδομένα:

- *"private Player pawn_owner"* Ο ιδιοκτήτης του πιονιού.
- *"private boolean is_theseus"* Boolean η οποία είναι TRUE αν είναι πιόνι-Θησέας και FALSE αν είναι αρχαιολόγος.
- *"private Position pawn_position"* Η θέση στην οποία είναι τοποθετημένο το πιόνι.
- *"private File pawn_image"* Η εικόνα του πιονιού.

Μέθοδοι:

- *"public Pawn(Player newOwner, boolean theseus_or_not, File newImage)"* Ο constructor της κλάσης.
- *"public void set_pawn_position(Position newPosition)"* Θέτει καινούργια θέση για το πιόνι.
- *"public Player getOwner()"* Επιστρέφει τον ιδιοκτήτη του πιονιού.
- *"public boolean get_is_theseus()"* Επιστρέφει τη boolean που υποδεικνύει αν το πιόνι αυτό είναι Θησέας.
- *"public Position get_pawn_position()"* Επιστρέφει τη θέση του πιονιού.
- *"public File get_pawn_image()"* Επιστρέφει την εικόνα του πιονιού.
- "*public String toString()"* Πληροφορίες σχετικά με το πιόνι.

❖ **Player**

Η κλάση Player αφορά κάθε ένα από τους 2 παίκτες του παιχνιδιού.

Δεδομένα:

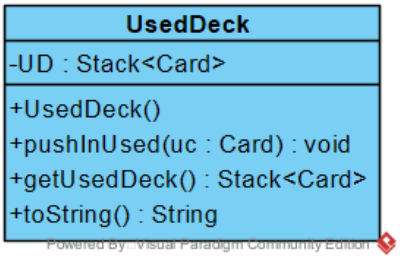
- *"private int player_score"* Το score του παίκτη.
- *"private ArrayList<Finding> list_findings"* Λίστα με τις αρχαιότητες που έχει ανακαλύψει ο παίκτης.
- *"private Hand player_hand"* Το χέρι με τα 8 φύλλα του παίκτη.
- *"private ArrayList<Card> last_cards_played"* Λίστα που αποθηκεύει το ιστορικό των καρτών που έχει παίξει ο παίκτης.
- *"private Pawn[] player_pawns"* Πίνακας με τα 4 πιόνια του παίκτη.

- *"private FrescoFinding[] player_pics"* Πίνακας με τις 6 φωτογραφίες από τοιχογραφίες του παίκτη. Αρχικοποιούνται και οι 6 στον πίνακα. Στο dialog "Οι Τοιχογραφίες μου" φαίνονται ως μη γκριζαρισμένες όσες έχουν ως ιδιοκτήτη τον συγκεκριμένο Player.
- *"private int player_snakegoddess_count"* Μετρητής των αγαλματιδίων της Θεάς των Φιδιών που έχει ανακαλύψει ο παίκτης.

Μέθοδοι:

- *"public Player()"* Ο constructor της κλάσης.
- *"public void set_player_score(int newScore)"* Θέτει καινούργιο score στον παίκτη.
- *"public int get_player_score()"* Επιστρέφει το score του παίκτη.
- *"public void add_new_finding(Finding newFinding)"* Τοποθετεί την αρχαιότητα newFinding στη λίστα με τις αρχαιότητες που έχει ανακαλύψει ο παίκτης.
- *"public ArrayList<Finding> get_list_findings()"* Επιστρέφει τη λίστα με τις αρχαιότητες που έχει ανακαλύψει ο παίκτης.
- *"public void player_draws_cards()"* Ο παίκτης τραβάει φύλλο από τη τράπουλα.
- *"public Hand getHand()"* Επιστρέφει το χέρι με τα 8 φύλλα του παίκτη.
- *"public void update_last_cards_played(Card c)"* Τοποθετεί τη κάρτα c στο τέλος του ιστορικού των φύλλων που έχει παίξει ο παίκτης.
- *"public ArrayList<Card> get_last_cards_played()"* Επιστρέφει το ιστορικό των καρτών που έχει παίξει ο παίκτης.
- *"public Pawn[] get_player_pawns()"* Επιστρέφει τον πίνακα με τα πιόνια του παίκτη.
- *"public void update_player_pics(FrescoFinding fresco_pic)"* Προσθέτει καινούργια φωτογραφία από τοιχογραφία στον πίνακα player_pics.
- *"public FrescoFinding[] get_player_pics()"* Επιστρέφει τον πίνακα με τις φωτογραφίες του παίκτη από τοιχογραφίες.
- *"public void update_player_snakegoddess_count()"* Προσθέτει ένα στον μετρητή των αγαλματιδίων της Θεάς των Φιδιών.
- *"public int get_player_snakegoddess_count()"* Επιστρέφει τη τιμή του μετρητή των αγαλματιδίων της Θεάς των Φιδιών.
- "*public String toString()"* Πληροφορίες σχετικά με τον παίκτη.

❖ UsedDeck



Η κλάση UsedDeck αφορά τη τράπουλα με τις κάρτες οι οποίες έχουν παιχτεί.

Δεδομένα:

- *"private Stack<Card> UD"* Στοίβα η οποία αποθηκεύει τα χαρτιά που έχουν χρησιμοποιηθεί.

Μέθοδοι:

- *"public UsedDeck()"* Ο constructor της κλάσης.

- *"public void pushInUsed(Card uc)"* Προσθέτει τη κάρτα uc στη κορυφή της στοίβας με τα χρησιμοποιημένα χαρτιά.
- *"public Stack<Card> getUsedDeck()"* Επιστρέφει τη στοίβα με τα χρησιμοποιημένα χαρτιά.
- "*public String toString()"* Πληροφορίες σχετικά με τη τράπουλα των χρησιμοποιημένων καρτών.

3. Η Σχεδίαση και οι Κλάσεις του Πακέτου Controller

❖ **Controller**

Δεδομένα:

- *"private int turn"* Ακέραια μεταβλητή που κρατάει τη σειρά των παικτών. Για turn=1 παίζει ο player1 και για turn=2 παίζει ο player2.
- *"private Player player1"* Ο πρώτος παίκτης.
- *"private Player player2"* Ο δεύτερος παίκτης.
- *"private Board gameBoard"* Το ταμπλό του παιχνιδιού.
- *"private GraphicUI gameWindow"* Το παράθυρο με τα γραφικά του παιχνιδιού.

Μέθοδοι:

- *"public Controller()"* Ο constructor της κλάσης.
- *"public void yourTurn()"* Δίνει την σειρά στον επόμενο παίκτη.
- *"public int getTurn()"* Επιστρέφει τη τιμή του ακεραίου που αποθηκεύει τη σειρά των παικτών.
- *"public void initializePlayer(Player p)"* Αρχικοποιεί τον παίκτη p.
- *"public Player getPlayerPlaying()"* Επιστρέφει τον παίκτη που παίζει τη δεδομένη στιγμή.

- *"public void initializeGameBoard(Board b)"* Αρχικοποιεί το ταμπλό του παιχνιδιού.
- *"public Board getGameBoard()"* Επιστρέφει το ταμπλό του παιχνιδιού.
- *"public void initializeGameWindow(GraphicUI gw)"* Αρχικοποιεί το παράθυρο γραφικών του παιχνιδιού.
- *"public void updateGameWindow(GraphicUI gw)"* Ανανεώνει το παράθυρο γραφικών του παιχνιδιού.
- *"public GraphicUI getGameWindow()"* Επιστρέφει το παράθυρο γραφικών.
- *"public boolean isGameFinished(Board b)"* Ελέγχει αν έχει λήξει το παιχνίδι βάσει των κανόνων και επιστρέφει TRUE αν έληξε και FALSE αν όχι.
- *"public Player findWinner()"* Αν έχει λήξει το παιχνίδι ψάχνει και επιστρέφει τον νικητή.
- *"public void announceWinner(Player p)"* Ανακοινώνει τον νικητή του παιχνιδιού ή ισοπαλία.
- *"public static void main(String[] lala)"* Η main του project. Εδώ εκτυλίσσεται η βασική λούπα του παιχνιδιού.
- "*public String toString()"* Πληροφορίες σχετικά με τον controller.

4. Η Σχεδίαση και οι Κλάσεις του Πακέτου View

Σε αυτό το packet βρίσκεται η κλάση GraphicUI για το βασικό panel του παιχνιδιού, εντός του οποίου υπάρχει ένα pane για κάθε παίκτη, το κουμπί "Οι Τοιχογραφίες μου", το κουμπί της τράπουλας και τα 16 κουμπιά των φύλλων και των δύο παικτών. Επίσης, έχω ορίσει ένα constructor αυτής της κλάσης, μια μέθοδο *"public void initComponents()"* που αρχικοποιεί κάθε panel και κουμπί, καθώς και ένα ActionListener για τα φύλλα των παικτών με όνομα *PlayListener.*

Τέλος, εντός του πακέτου View υπάρχουν και οι κλάσεις *FindingDialog* και *FrescoDialog*, οι οποίες αφορούν τα καινούρια παράθυρα που ανοίγουν κάθε φορά που ένας παίκτης ανακαλύπτει ένα *RareFinding* και όταν ο παίκτης πατάει το κουμπί "Οι Τοιχογραφίες μου" αντίστοιχα. Στο *FindingDialog* (μέχρι στιγμής) υπάρχει το string με το κείμενο που παρουσιάζει, η εικόνα της αρχαιότητας, ένας constructor και μια μέθοδος *"public String choice()"* η οποία επιστρέφει την επιλογή του χρήστη. Ομοίως, στο *FrescoDialog* (μέχρι στιγμής) υπάρχει ένας πίνακας με τις φωτογραφίες των τοιχογραφιών, ένας constructor και η ίδια μέθοδος *"public String choice()"* η οποία επιστρέφει την επιλογή του χρήστη.

Παπαδάκης Γεώργιος 4975