Classe Card

Attributi

1. cardID: int
2. player: Player (settato di default a null poiché inizialmente non appartiene a nessun giocatore)
3. side: boolean (settato di default a 0 poiché coperta inizialmente, non serve un campo apposito nel costruttore)

1. ID identifica univocamente la carta in fase di memorizzazione
2. attributo al giocatore che possiede la carta, modificato dal controller una volta posseduta nella mano del giocatore e utile per i metodi attribuzione punti
3. ogni carta ha, da regolamento due side, la scelta di un boolean è dovuta alla scelta preventiva di una corrispondenza biunivoca fra 1, il FRONT, 0 il BACK.

   Il FRONT e il BACK sono definiti dal file fornito dal professore, non dal regolamento (poiché sono contradditori). Il BACK è il lato comune a tutte le carte poiché' in fase di distribuzione non si devono distinguere, il FRONT varia da carta a carta.

Gli oggetti di tipo Card si suddividono in due sottoclassi: **PlayingCard** (carte dedite al posizionamento), **ObjectiveCard**(carte dedite al conseguimento di un obbiettivo per ricevere punti, non giocabili sul board)

CONVENZIONE: i corner sono rappresentati da un ArrayList, ordinati, indipendentemente dal lato scelto:
Indice0, alto sinistra
Indice1, alto destra
Indice2, basso sinistra
Indice3, basso destra

Metodi

- costruttore
- metodi getter
- metodi setter

Classe PlayingCard – sottoclasse di Card

Attributi

1. resourceCornerFront: RESOURCES [4] (boolean side ==1)
2. resourceCenterBack: RESOURCES
3. positionX: int
   positionY: int
4. strat: **GoldCardStrategy**

1. I Corner nel FRONT (boolean side ==1) sono presenti in tutte le carte PlayingCard ma hanno contenuto diverso, perciò, vanno istanziati alla creazione della carta

2. I Corner nel BACK (boolean side ==0) sono presenti in tutte le carte PlayingCard ma hanno contenuto RESOURCE.free (guardare enumerazione RESOURCES)

3. Ogni PlayingCard puo' essere posizionata in un board a matrice, questo attributo associa univocamente la posizione (due coordinate) della carta una volta posizionata. Una volta creato i due attributi sono istanziati a un valore intero non appartenente alla matrice. X e Y inizializzate a -1

   DIPENDENZE: Le coordinate a cui vengono istanziate le carte inizialmente, prima di essere posizionate nel board, sono definite in base a come viene definita il board e quali quadranti del piano cartesiano si usano.

4. Riferimento all'interfaccia GoldCardStrategy, implementata dalle 3 classi GoldCardBasic, GoldCardResource, GoldCardCorners, che ridefiniscono l'assegnamento punti in base al tipo di carta. → Sarà *null* per alcune carte risorsa e tutte le carte iniziali

Metodi

- costruttore
- metodi getter
- metodi setter

Classe ResourceCard – sottoclasse di PlayingCard

Attributi

1. resourceCornerBack: RESOURCES [4]
2. points: int

    1. sempre settato a free, perché il retro di una carta non ha risorse sugli angoli
    2. alcune ResourceCard attribuiscono 1 punto una volta posizionate (campo aggiunto al costruttore)

Metodi

- costruttore
- metodi getter
- metodi setter
- addPoints(), per aggiungere i punti al personalScore dell'owner

Classe GoldCard – sottoclasse di PlayingCard

Attributi

1. resourceCornerBack: RESOURCES[4]
2. requirements: hashMap <RESOURCES, int>
3. points: int
4. strat: **GoldCardStrategy**

    1. sempre settato a free, perché il retro di una carta non ha risorse sugli angoli (lo abbiamo messo solo perché semplifica i metodi della personalBoard)

    2. Il posizionamento della carta è possibile solo se si verifica il possedimento di risorse indicate sulla carta e differenti di carta in carta, creiamo una mappa con chiave la risorsa e un intero per il numero di risorse da possedere per poterla posizionare. Tali attributi vanno specificati nel costruttore.

    3. Tutte le GoldCard attribuiscono (x) punti una volta posizionate oppure per il completamento di un obbiettivo. (campo punti aggiunto al costruttore)

    4. Riferimento all'interfaccia GoldCardStrategy, implementata dalle 3 classi GoldCardBasic, GoldCardResource, GoldCardCorners, che ridefiniscono l'assegnamento punti in base al tipo di carta

Metodi

- costruttore
- metodi getter
- metodi setter
- addPoints(), per aggiungere i punti al personalScore dell'owner, con rispettivi override

Classe InitialCard – sottoclasse di PlayingCard

Attributi

1. resourceCenterFront: hashMap<RESOURCES, int>
2. resourceCornerBack: RESOURCES[4]

    1. le carte iniziali sono le uniche ad avere un centro sul FRONT e niente sul BACK, con più di una risorsa. Campi da inizializzare sul costruttore.

    2. usando la convenzione del lato di BACK esposta nei punti precedenti, la carta iniziale è l'unica ad avere i Corner sul BACK con presente una risorsa, tali risorse non sono fisse, perciò, serve inserire tali campi nel costrutto

Metodi

- costruttore
- metodi getter
- metodi setter

classe ObjectiveCard – sottoclasse di Card

1. points: int
2. strat: ObjectiveCardStrategy

    1. tutte le ObjectiveCard attribuiscono (x) punti a fine partita per ogni volta che l'obbiettivo è stato completato

    2. Riferimento all'interfaccia ObjectiveCardStrategy, implementata dalle 3 classi ResourceStrat, DPatternStart, LPatternStrat, che ridefiniscono l'assegnamento punti in base al tipo di obiettivo

Metodi

- costruttore
- metodi getter
- metodi setter
- addPoints(), con rispettivi override

Classe Deck

Attributi

1. card1: Card
2. card2: Card
3. deck [0...40]: ArrayList<Card>

    1. costruito il Deck è inizializzata a vuoto e solo dopo chiamata è lo spazio destinato alle carte scoperte, da regolamento c'è una prima carta comune per le ResourceCard, GoldCard, ObjectiveCard.

    2. costruito il Deck è inizializzata a vuoto e solo dopo chiamata è lo spazio destinato alle carte scoperte, da regolamento c'è una seconda carta comune per le ResourceCard, GoldCard, ObjectiveCard

    3. Mazzo costituito dalle carte da cui pescare o distribuire carte.

Metodi

- costruttore
- metodi getter
- metodi setter

Classe DeckBoard

Attributi

1. goldDeck: Deck
2. resourceDeck: Deck
3. objectiveCard1: ObjectiveCard
4. objectiveCard2: ObjectiveCard

    1. Deck costituito da carte di tipo GoldCard
    2. Deck costituito da carte di tipo ResourceCard
    3. Obiettivo comune 1
    4. Obiettivo comune 2

Metodi

- costruttore
- metodi getter
- metodi setter

Il resto delle classi è commentato sull'UML