DESCRIZIONE UML PROGETTO INGEGNERIA DEL SOFTWARE

MEMBRI DEL GRUPPO

- Polizzi Chiara
- Raimondi Francesco
- Rubagotti Andrea
- Santagata Maria Concetta (REFERENTE)

FUNZIONALITÀ AGGIUNTIVE

Sono presenti classi anche per le 2 funzionalità aggiuntive che abbiamo deciso di implementare (chat, partite multiple).

DESCRIZIONE UML

N.B: L'UML è di alto livello, di conseguenza è privo di tutti i metodi getter e setter di ogni classe in esso presente (li consideriamo sottintesi).

La classe **Game** si occupa di gestire l'intera partita dall'inizio (distribuzione di tutte le carte iniziali, gestione delle **Pawn** (pedine) del singolo giocatore, ecc.) alla fine (controllo del vincitore con relativi conteggi dei punti). Ogni Game è contraddistinto da uno stato di gioco (**GameState**), il quale può assumere diversi valori in base alla fase in cui lo stesso si trova: started, ending (settato appena un giocatore raggiunge 20 punti o entrambi i mazzi di pesca terminano), ended, waiting for start).

La classe Game contiene una lista di **Chat** (che possono essere da Player a Player, oppure un'unica chat comprendente tutti i membri della partita). Ogni chat è composta a sua volta da una lista di **Message**, che contengono in una stringa il testo scambiato dai giocatori attraverso la piattaforma di gioco.

Ogni Game è composto anche da una lista formata da 2 a 4 **Player**. Questi ultimi sono implementati in modo da avere funzionalità di pesca della carta (dai mazzi o dalle carte già scoperte sul tavolo) e funzionalità per posizionare (giocare) una carta in una specifica posizione sulla Board. Inoltre ogni Player è contraddistinto da un suo stato (**PlayerState**) che può essere: is playing, is waiting oppure is disconnected e da una propria pedina (**Pawn**).

Ogni Player è legato ad una **Board** la quale è composta principalmente da una tabella (matrice) che contiene tutte le carte (Playable Card) presenti sul tavolo del giocatore ad essa relativo. La dimensione della matrice viene calcolata dinamicamente in base al numero di giocatori che formano la partita (da 2 a 4) e al numero di carte giocabili, che in questo gioco sono sempre 80. Dentro la board ci sono due set di **Coordinates** (formate da int x, int y) che permettono di stabilire dove il giocatore possa effettivamente giocare una carta, la mappa di playedCards, invece, ci consente nella fase finale della partita di calcolare i punti ottenuti completando gli obiettivi "pattern" delle carte obiettivo.

Per quanto concerne la gestione delle carte, la classe **Card** è una classe astratta che contiene un ID e abbiamo da questa due classi figlie: Playable e Objective.

- La classe **Playable Card** rappresenta l'insieme delle carte oro, risorsa e base (da qui il grande numero di attributi). Le Playable Cards (risorsa, oro, base) sono raggruppate in diverse istanze di **Deck** (mazzi) che sono degli Stack (quindi offrono metodi di libreria).
- Le **Objective Card** sono suddivise in due sottoclassi: quelle i cui punti dipendono da risorse/oggetti presenti sulla board (**Objective_card_number**), e quelle che invece

attribuiscono punti solo se vengono rispettati specifici “pattern” nel piazzamento delle carte (**Objective_card_pattern**). Le due sottoclassi ridefiniscono (Overriding) il metodo addPointsToPlayer (player: Player), il quale si occupa di calcolare i punti derivati dalla singola carta obiettivo e li assegna automaticamente al giocatore. Le carte obiettivo, a differenza delle Playable cards, non vengono raggruppate in un Deck, ma in un array di 16 posizioni.

Non sono presenti nell’UML tutti i metodi che utilizziamo per leggere i file JSON per inizializzare tutte le carte del gioco all’inizio della partita.

Sono presenti tre tipi di *enumerazioni*:

- **AngleType**: contiene tutte le tipologie di angoli possibili (un angolo può anche essere vuoto, o può anche non essere presente)
- **CountingType**: contiene tutte le tipologie di elementi che devono essere “contati” per poter raggiungere gli obiettivi di risorse o per poter posizionare le carte oro
- **CentralType**: contiene solamente le quattro risorse principali, che possono essere presenti al centro della carta (fungi, animal, insect, nature).