PROGETTAZIONE GIOCO DA TAVOLO “MONOPOLY”

DESCRIZIONE: Programma che permette di simulare una partita con più giocatori locali a monopoly.

-REGOLE:

- Si può giocare fino a 4 giocatori

- Vince l’ultimo che ha soldi disponibili o il più ricco dopo un certo numero di turni

- La quantità di soldi iniziale di ogni giocatore viene decisa in base al numero di partecipanti

- I giocatori iniziano senza alcuna proprietà

- Durante il proprio turno, il giocatore può richiedere uno scambio ad un altro giocatore

- Tutte le azioni che il giocatore può eseguire devono essere fatte prima del tiro dei dadi

FUNZIONALITA’:

- I giocatori possono essere sia umani che controllati dal computer

- L’utente può decidere se la partita debba continuare finchè rimane un solo giocatore con dei soldi, o se la partita debba finire dopo un determinato numero di turni

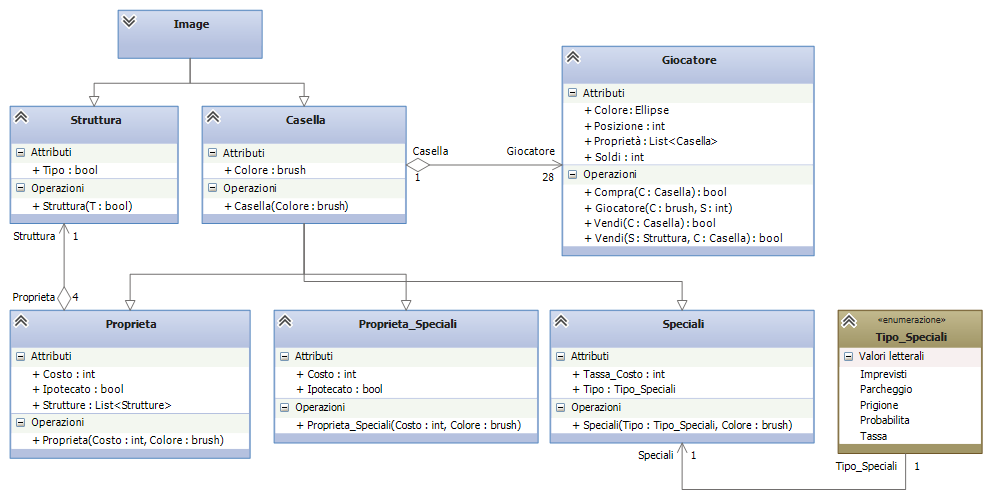
- La posizione iniziale della pedina è già definita e non può essere cambiata

- L’ordine del turno è in senso antiorario (Non viene scelto tirando un dado all’inizio)

- Le pedine si muovono automaticamente a seconda del punteggio dei dadi ottenuto

- Una volta tirati i dadi, il turno del giocatore corrente finisce e si passa a quello successivo

*Classi*



**Giocatore**: E’ distinto dal proprio colore, una lista di tutte le proprietà che possiede e un numero di soldi, che servono anche a verificare se l’utente può continuare a giocare o ha perso.

\*Colore: Colore del giocatore

\*Posizione: Posizione corrente del giocatore sul tabellone

\*Proprietà: Lista delle proprietà che il giocatore possiede

\*Soldi: Quantità di denaro posseduta dal giocatore

+ Giocatore(C: brush, S: int) - Costruttore della classe giocatore, assegna il colore allo stesso e il numero di soldi iniziali, che variano a seconda del totale di partecipanti.

+ Compra(C: Casella): bool - Metodo che viene richiamato quando il giocatore vuole acquistare la casella su cui è. Restituisce TRUE se il giocatore ha abbastanza soldi per comprare la proprietà, FALSE il contrario.

+ Vendi (C: Casella): bool – Metodo che viene richiamato quando il giocatore decide di vendere (ipotecare) una proprietà. Se la proprietà appartiene allo stesso giocatore che la vuole vendere, ritorna TRUE. Se si sta cercando di vendere una proprietà di qualcun altro, non verrà venduta (FALSE).

+ Vendi (S: Struttura, C: Casella) – Questo metodo serve per vendere le strutture (case o alberghi che siano) costruite sulla proprietà. Allo stesso modo, il giocatore può sempre vendere strutture (ammesso che ce le abbia).

**Struttura : Image**: Questa classe serve solamente per distinguere il tipo di struttura che si vuole utilizzare per compiere un’operazione ( Casa o albergo ).

\*Tipo: Tipo di struttura, casa o albergo

+ Struttura(T: bool) – Costruttore della classe che ritorna TRUE se la struttura è una casa, FALSE se è un albergo

**Casella**: Classe astratta, ereditata per il colore delle caselle.

\*Colore: Colore della casella

\*Nome: Nome della casella

+ Casella (Col: brush, N: string) – Costruttore, assegna il colore alla casella e il suo nome.

**Proprieta : Casella**: Classe che serve per istanziare una proprietà normale con un costo. La proprietà potrà essere ipotecata dal proprietario, e quest’ultimo potrà costruirci sopra le strutture.

**\***Costo: Prezzo della proprietà

\*Ipotecato: Variabile che segna se la proprietà è stata ipotecata (venduta temporaneamente)

\*Proprietario: Giocatore che ha comprato la proprietà analizzata

\*Speciale: Variabile che segna se la proprietà non è acquistabile (Prigione, parcheggio, imprevisti, ecc.)

\*Strutture: Lista di strutture costruite sulla proprietà

+ Proprieta(Col: brush, N: string, C: int, S: bool) – Costruttore, assegna il colore, il nome, il costo e se è speciale (stazione o società elettrica/idrica) alla proprietà.

**Speciali : Casella**: Classe che serve per istanziare le proprietà speciali, come la prigione, le carte di imprevisti o probabilità. Avrà un tipo di casella speciale (come quelli riportati prima) e, se si tratta di una tassa, avrà anche l’importo da pagare.

\*Tassa\_Costo: Tassa da pagare quando si finisce sopra ad una proprietà speciale. Se non è una tassa, il costo sarà zero.

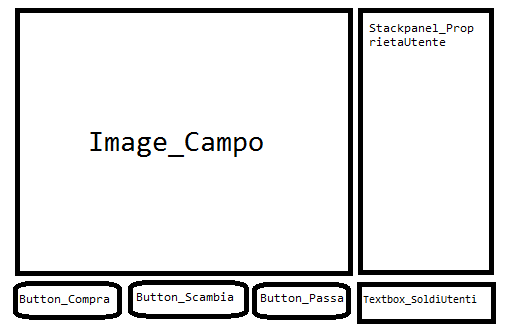
\*Tipo: Genere di proprietà speciale

+ Speciali(N: string, T: Tipo\_Speciali, C: int): Costruttore, viene assegnato il suo nome, il tipo di proprietà speciale e l’importo da pagare quando ci si finisce sopra (nel caso sia una tassa).

**Tipo\_Speciali**: Enumeratore che definisce il tipo di proprietà speciale (Prigione, imprevisti, ecc.)

View

Il tabellone di gioco viene rappresentato da un’immagine, sulla quale si sposteranno le pedine dei giocatori, al centro della finestra del programma. Sotto di esso ci saranno tre pulsanti, che rappresentano tutte le azioni possibili che il giocatore può effettuare durante il suo turno. Quando il giocatore tira il dado, il turno passerà al giocatore successivo. Alla destra della finestra saranno visualizzate in una stackpanel tutte le proprietà che il giocatore possiede, e al di sotto di essa saranno visualizzati i soldi di tutti i giocatori



**Image\_Campo**: Immagine del campo su cui si muoveranno le pedine.

**Stackpanel\_ProprietaUtente**: Stackpanel che mostrerà le proprietà di tutti giocatori

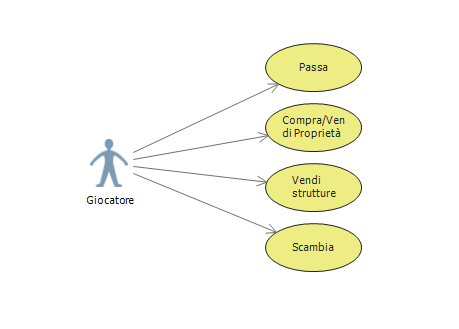
**Textbox\_SoldiUtenti**: Textbox che tiene conto dei soldi di tutti i giocatori.

**Button\_Compra**: Bottone che serve per comprare la proprietà su cui si è (se si può comprare) o per vendere le proprietà o strutture desiderate dagli utenti senza contare la casella su cui sono.

**Button\_Scambia**: Bottone che serve a proporre uno scambio ad un altro giocatore.

**Button\_Passa**: Bottone che farà avanzare le pedine dei giocatori, per poi passare il turno.

Casi d’uso



*Sequenza del programma*

