Classe: capaVistes

Funcionament resumit de forma pseudocodi:
No prengeu al peu de la lletra, NO HI HAURAN aquests bucles, per si de cas algú s'imagina el que no és.

```
novaPartida (llistat_jugadors_ordenats, diners_inicials)
Mentre No fiPartida()
 Mentre No fiMà()
  DibuixarPantalla (fent ús de obtenirCartesX)
  {participant_actual,min,max} = properParticipant()
  si esHuma()
   DemanarAccióUsuari(participant_actual,min,max)
   si EsRetira retiraParticipant()
   else apostaParticipant(diners_que_aposta)
  else
   resultat = jugadorNoHumàJuga()
   MostraAlUsuariJugadaDeLaMàquina(resultat)
   si EsRetira retiraParticipant()
   else apostaParticipant(diners_que_aposta)
  fsi
 fiMentre
  MostraGuanyadorMà()
fiMentre
MostraGuanyadorPartida()
```

novaPartida (string [] jugadors, int diners, int apostaMàxima, int MaxMans)

Crea una nova partida donada una llista de noms de jugadors que participaran així com els diners inicials de cada participant. Descarta la partida que podia estar en joc anteriorment. També l'aposta màxima o el nombre màxim de mans, que pot ser infinit (no limit).

properParticipant () string jugador, int aposta_mínima, int aposta_màxima

Indica al sistema que passi al següent jugador (assumint que l'actual ja ha jugat). El sistema retorna qui és el següent jugador (nom) així com l'aposta mínima que ha de fer per a continuar i l'aposta màxima que pot fer. *Això és molt flexible, ja que podem afegir límits al joc i alhora jugar la cega de forma transparent a la capa de vistes.*

apostaParticipant (int diners)

El participant actual (l'últim que ha retornat properParticipant òbviament) aposta la quantitat diners, que estarà entre min i max.

retiraParticipant ()

El participant actual es retira.

obtenirCartesParticipant (string jugador) Carta []

Obté les cartes que té el jugador amb nom "jugador" per a poder fer la seva visualització si s'escau. El format de Carta cal definir-lo ja que no crec convenient usar cap classe de la capa domini encara que el profe digués que se podia fer. El més senzill és que sigui una merda de tupla ja que no ha de fer res més que informar.

obtenirCartesTaula () Carta []

Obté les cartes de la taula per a poder fer dibuixets i tal.

fiPartida () bool, jugador_guanyador

Retorna si ha acabat el joc i qui l'ha guanyat.

fiMà () bool, jugador_guanyador

Retorna si ha acabat la mà i tal. Important pel tema de informar als users. Retorna qui l'ha guanyada.

ÉsHumà () bool

Retorna si el jugador actual és un jugador humà.

jugadorNoHumàJuga () acció, diners

Retorna l'acció que fa el jugador no humà en aquest moment i els diners que aposta si s'escau.

LlistaParticipants () string jugador, string avatar, string password, bool huma, ...

Retorna el llistat de jugadors (tots! No només

participants) amb les seves “propietats”. Cal dir que és necessari per a mostrar avatars i per a triar qui jugarà la partida a l'inici entre d'altres coses.

La “caspa” de domini inclourà aquestes funcions, clar. És un tant absurd que hi hagi una crida a vistes i aquesta cridi a domini però es fa així ves. En principi són bastant iguals les dues capes vistes/domini i les funcions principals són aquestes. Ara cal dir quines funcions més tindrà la capa de domini entre les que hi haurà les decisions de la IA, els càlculs de guanyador així com els d'aposta màxima mínima i un llarg etc.

Què us sembla, machaqueume!!!! A mi me pone!

Guia d'implementació del programa.

Funcions de la capa domini

Nota: He omès l'objecte pòquer. Potser no és ni necessari. M'he referit a partida_actual com a l'objecte que representa la única partida en joc.

```
novaPartida (string [] jugadors, int diners, int apostaMàxima, int
MaxMans) {

 Partida partida_actual = new Partida(jugadors, diners,
                              apostaMàxima, MaxMans);

}

properParticipant () string jugador, int aposta_mínima, int
aposta_màxima {

 Participant pa = partida_actual.obtenirParticipantActual();

 int mínima = partida_actual.obtenirMínimaApostaRonda();
 mínima = mínima - pa.getApostaAct();
 if (mínima > pa.getDiners()) mínima = pa.getDiners();
          //El jugador no té diners per igualar

 int màxima = max(pa.getDiners(),
                  partida_actual.getApostaMàxima());

 retorna { pa.getJugador().getNom(),
           mínima,
           màxima };

}

apostaParticipant (int diners) {
 Participant pa = partida_actual.obtenirParticipantActual();

 partida_actual.apostaParticipant(diners, pa);
}

retiraParticipant (int diners) {
 Participant pa = partida_actual.obtenirParticipantActual();

 partida_actual.retiraParticipant(diners, pa);
}

obtenirCartesParticipant (string jugador) Carta [] {
  Participant [] pas = partida_actual.getParticipants();

  for each Participant pa in pas
   if (pa.getJugador().getNom() == jugador) {
```

```
    MàDeCartes mdc = pa.getCartes();
    break;
   }
  }

  retrona mdc.getCartes();  //Conversió necessària per vistes
}

obtenirCartesTaula () Carta [] {
 retorna partida_actual.getCartesTaula();  // Conversió
 // un objecte de la capa domini no ha de passar a vistes!
}
```