Projecte: Escacs
Segona part

Metodologia de la Programació

Curs 2021 - 2022

Recordem... Planificació del projecte

El projecte el desenvoluparem en dues fases, que es correspondran amb el lliurament parcial i el lliurament final del projecte.

Primera versió del projecte:

- Inicialitzar el tauler del joc a partir de la informació guardada a un fitxer de text.
- Determinar els moviments vàlids de qualsevol peça del tauler.
- Moure una peça, comprovant que el moviment és vàlid.
- Mostrar l'estat actual del tauler.
- En aquesta primer versió treballarem sense visualització gràfica. Tindreu un test d'autoavaluació a Caronte per poder validar el correcte funcionament de les diferents funcionalitats.

Segona versió del projecte:

- Implementar la part gràfica del joc i la interacció del jugador amb el tauler durant el seu torn.
- Implementar el desenvolupament complet d'una partida a partir d'un estat inicial, alternant els torns dels jugadors fins al final de la partida.
- Guardar en un fitxer els moviments que es fan durant el desenvolupament de la partida.
- Reproduir una partida prèviament jugada executant els moviments guardats en un fitxer.

Segona versió del projecte

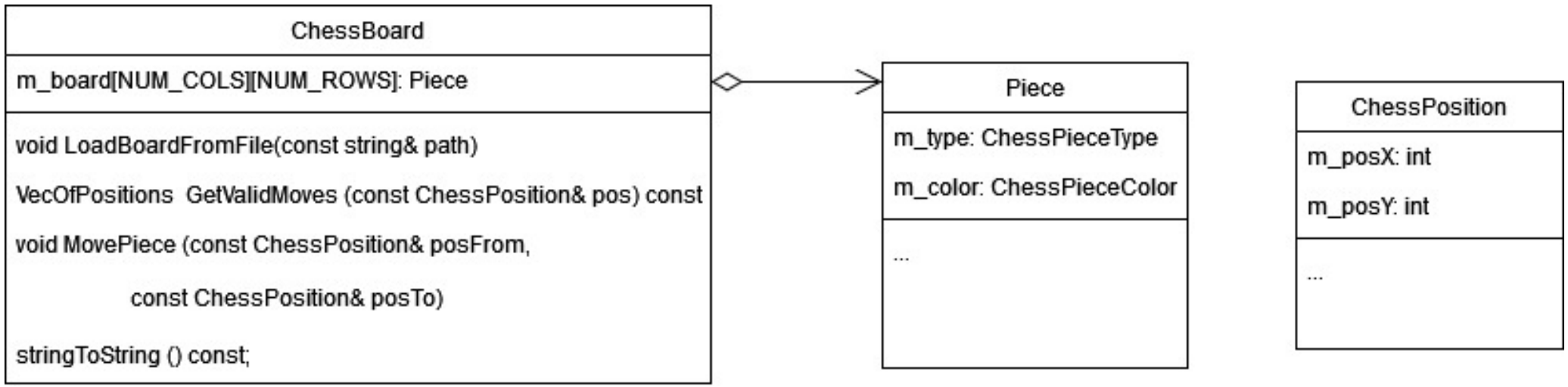
Noves funcionalitats

- Implementar la part gràfica del joc i la interacció del jugador amb el tauler durant el seu torn:
  - Visualitzar gràficament el tauler de joc amb totes les peces a les seves posicions actuals.
  - Permetre seleccionar amb el ratolí una peça a moure i mostrar gràficament les posicions vàlides a on es pot moure aquesta peça.
  - Seleccionar amb el ratolí la posició destí a on volem moure la peça i si és una de les posicions vàlides moure la peça a la nova posició.
- Implementar el desenvolupament complet d'una partida a partir d'un estat inicial, alternant els torns dels jugadors fins al final de la partida.
  - Mostrar per pantalla quin jugador té el torn de joc en cada moment.
- Guardar en un fitxer tots els moviments que es fan durant el desenvolupament de la partida.
- El joc haurà de tenir dos modes d'execució:
  - Jugar una partida normal
  - Reproduir una partida prèviament jugada executant els moviments guardats en un fitxer.

Mireu el vídeo que hi ha a Caronte que mostra el funcionament de la partida

Recordem... Primera versió del projecte

Estructura de classes



Noves classes

CurrentGame
m_board: Board m_movements: QueueMovements ...

void init(GameMode mode,
  const string& intitialBoardFile,
  const string& movementsFile)
void updateAndRender(int
  mousePosX, int mousPosY,
  int mouseStatus)
void end()

Classe `CurrentGame`

- Guarda tota la informació necessària per gestionar una partida:
  - El tauler de joc amb un objecte de la classe Chessboard
  - La cua amb tots els moviments que s'han fet des de l'inici amb un objecte de la classe QueueMovements
  - Altres atributs necessaris per gestionar la partida: mode de joc de la partida (normal o reproduint moviments d'un fitxer), jugador amb el torn actual, si s'ha arribat al final de la partida i qui és el guanyador, si hi ha alguna peça seleccionada per moure i quina és, llista de moviments vàlids de la peça seleccionada, ...
- Us suggerim que tingui com a mínim aquests mètodes:
  - `init`: s'encarrega d'inicialitzar la partida segons el mode de joc.
  - `updateAndRender`: s'encarrega de la interacció amb el jugador i la visualització de l'estat actual de la partida per pantalla.
  - `end`: s'encarrega de les accions que s'han de fer quan s'acaba una partida.

Noves classes

GameInfo.h

```
typedef enum {
    GM_NORMAL,
    GM_REPLAY,
    GM_NONE
} GameMode;
```

CurrentGame
m_board: Board m_movements: QueueMovements ...
void init(GameMode mode, const string& intitialBoardFile, const string& movementsFile) bool updateAndRender(int mousePosX, int mousPosY, int mouseStatus) void end()

Classe CurrentGame: mètodes

```
void init(GameMode mode, const string& intitialBoardFile, const string& movementsFile);
```

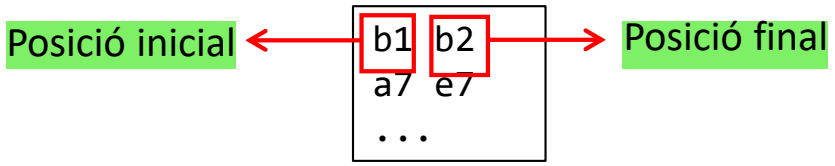
- Inicialitza el tauler llegint les posicions inicials de les peces del fitxer indicat al paràmetre `initialBoardFile`.
- Si el mode és `GM_NORMAL`:
  - El fitxer indicat al paràmetre `movementsFile` s'haurà d'utilitzar per guardar tots els moviments fets al final de la partida.
- Si el mode és `GM_REPLAY`:
  - S'haurà d'inicialitzar la cua de moviments amb la informació del fitxer indicat al paràmetre `movementsFile`

Format fitxer `initialBoardFile`

```
0. Rb1
0. Da7
...
1. Pb6
1. Pc7
```

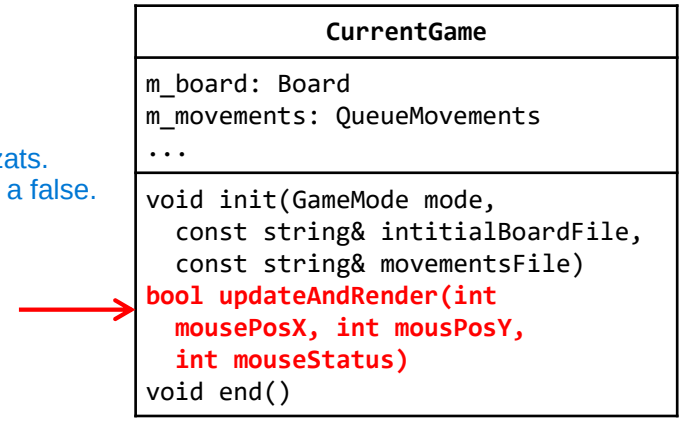
Igual que a la primera part del projecte

Format fitxer `movementsFile`



Noves classes

esta clase també ha de indicar si el jugador vol desfer un moviment, es a dir,
ha clicat el ratolí en unn espai on possi 'desfer'
actualitzara la pila de moviments eliminants l'ultim moviment i cambiará el
bool enrere a true per a que el init tregui el moviment del txt de moviments realitzats.
una vegada el init tregui el moviment del txt, ha de tornar a possar el bool enrere a false.



Classe CurrentGame: mètodes

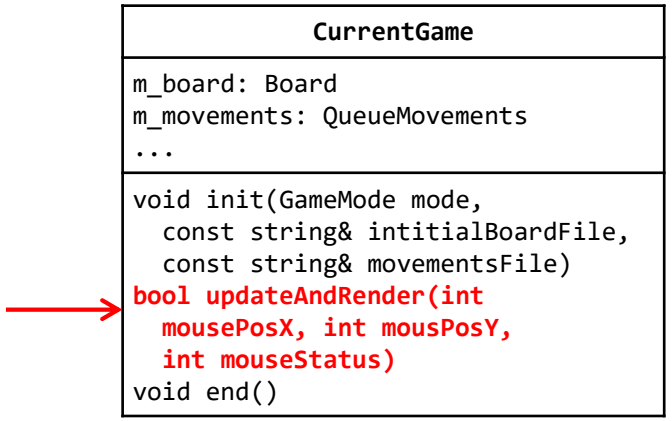
```
bool updateAndRender(int mousePosX, int mousePosY, bool mouseStatus);
```

- Si el mode és GM_NORMAL:
  - Si s'ha fet clic amb el ratolí a sobre d'una peça del color del jugador que té el torn, aquesta peça ha de quedar seleccionada com la peça que es vol moure i s'hauran de recuperar els moviments vàlids que pot fer la peça.

    Quan es visualitzi el tauler i les peces, s'hauran de mostrar les posicions dels moviments vàlids superposant a cada casella un requadre transparent verd (gràfic IMAGE_VALID_POS)
  - Si s'ha fet clic amb el ratolí a una de les posicions dels moviments vàlids de la peça seleccionada s'haurà de moure la peça que estigui seleccionada a aquesta nova posició i guardar el moviment a la cua de moviments realitzats. També s'haurà de comprovar si s'ha acabat la partida perquè es mata el rei del jugador contrari i, si no s'acaba, canviar el jugador que té el torn.
  - Si no s'interactua amb el ratolí o es fa clic a sobre de qualsevol altra posició de pantalla, no s'ha de fer res, simplement visualitzar el tauler, les peces i les posicions dels moviments vàlids (si hi ha alguna peça seleccionada)

Mireu el vídeo que hi ha a Caronte que mostra el funcionament de la partida

Noves classes



Classe CurrentGame: mètodes

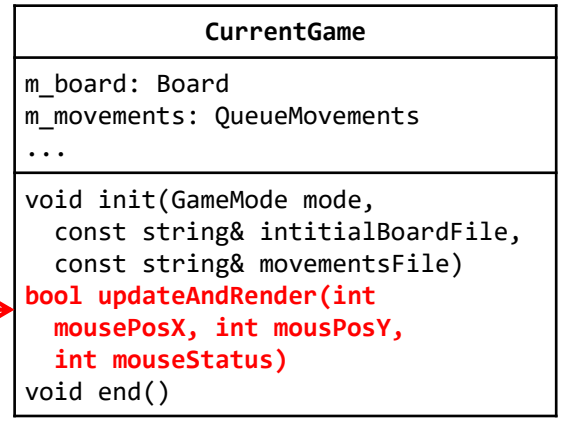
```
bool updateAndRender(int mousePosX, int mousePosY, bool mouseStatus);
```

- Si el mode és GM_REPLAY:
  - Si s'ha fet clic amb el ratolí a sobre del tauler de joc s'ha de recuperar i eliminar el primer moviment de la cua de moviments, executar-lo i visualitzar el nou estat del tauler i de les peces. També s'haurà de comprovar si s'ha acabat la partida perquè es mata el rei del jugador contrari i, si no s'acaba, canviar el jugador que té el torn.

    Si no queden moviments a la cua i encara no s'ha arribat al final de la partida s'haurà de mostrar un missatge per pantalla indicant que no es poden fer més moviments.
  - Si no s'interactua amb el ratolí o es fa clic a sobre de qualsevol altra posició de pantalla, no s'ha de fer res, simplement visualitzar el tauler i les peces.

Mireu el vídeo que hi ha a Caronte que mostra el funcionament de la partida

Noves classes



Classe CurrentGame: mètodes

```
bool updateAndRender(int mousePosX, int mousePosY, bool mouseStatus);
```

- En qualsevol dels dos modes, GM_NORMAL i GM_REPLAY:
  - En tot moment s'haurà de mostrar per pantalla en quin mode s'està jugant i quin és el jugador que té el torn actual.
  - Si la partida ja s'ha acabat s'haurà de mostrar per pantalla quin és el jugador que ha guanyat i retornar `true` com a resultat de la funció.

Mireu el vídeo que hi ha a Caronte que mostra el funcionament de la partida

Noves classes

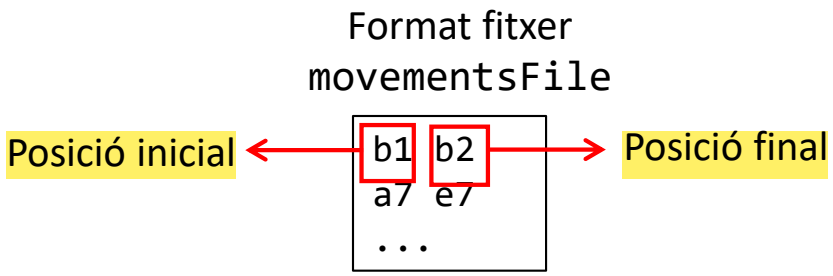
CurrentGame
m_board: Board m_movements: QueueMovements ...

void init(GameMode mode,
  const string& intitialBoardFile,
  const string& movementsFile)
bool updateAndRender(int
  mousePosX, int mousPosY,
  int mouseStatus)
void end()

Classe `CurrentGame`: mètodes

`void end();`

- Si el mode és GM_NORMAL:
  - S'haurà de guardar la cua de moviments al fitxer que s'ha indicat al paràmetre `movementsFile` quan s'ha fet la crida al mètode `init`
- Si el mode és GM_REPLAY, no s'ha de fer res.



Modificacions de classes existents

Classes `Piece` i `Chessboard`

- Us suggerim **afegir** a cadascuna d'aquestes dues classes un **mètode `render`** que permeti visualitzar els objectes per pantalla.
  - A la classe `Piece` aquest mètode hauria de dibuixar el gràfic de la peça (en funció del seu tipus i color) a la posició `x` i `y` que se li passi com a paràmetre.
  - A la classe `Chessboard` aquest mètode hauria de dibuixar el gràfic del tauler de joc i dibuixar també totes les peces a les seves posicions actuals, fent crides al mètode `render` de cadascuna de les peces del tauler. També es pot encarregar de visualitzar les posicions que corresponen als moviments vàlids de la peça seleccionada.
    Aquest mètode es cridarà des del mètode `updateAndRender` de la classe `CurrentGame` cada cop que es vulgui visualitzar el tauler.

Planificació del projecte

- **Lliurament final** del projecte: diumenge **12 de juny**
- Durant la setmana del **13 al 17 de juny** farem l'**avaluació** amb entrevistes **online** a cada grup per Teams.
- A la sessió de classe del dimecres **25 de maig** dedicarem una estona a resoldre **dubtes i preguntes** sobre el projecte.