

Univopoly

SOUTENANCE

1

Présentation projet

Qu'est ce que Univopoly ?

2

Vue d'ensemble

Présentation du
diagramme de classe

3

Corps du programme

Présentation du corps
du programme avec
illustration du mode
texte

4

Partie graphique

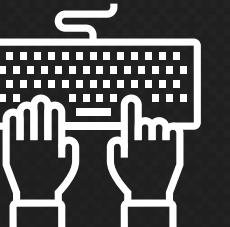
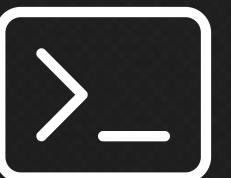
Partie consacrée à
l'explication de la
partie graphique
réalisée à l'aide de QT

Univopoly est un jeu de plateau inspiré du jeu Monopoly ou, peut-être est l'inverse ? L'univers du jeu est tiré de l'Université Claude Bertrand Lyon 1. Un jeu de plateau dans lequel évolue 1 à 8 joueurs où le but est d'user de stratagème capitaliste pour venir à bout des autres joueurs. Mais aussi aux joueurs riches de s'enrichir encore plus et au plus pauvre de se débrouiller !

Mode texte

Univopoly est disponible en mode texte.

On invite l'utilisateur à interagir avec le clavier pour jouer (de 1 à 8 joueurs).



Mode graphique

Un mode graphique est aussi implémenter les joueurs peuvent jouer avec le clavier et la souris et ainsi profitée pleinement de l'affiche graphique.



Corps du programme

Player	Inventory	Tile
<ul style="list-style-type: none"> - id: int - goods: Inventory - position: unsigned int - jail_count: int - number_gare: unsigned int + constructor, destructor + getId(): const int + getBalance(): int + getNetWorth(): int + transaction(IN amount unsigned int): boolean + getProperty(IN property_id unsigned int): like on Tile + buyProperty(IN lien sur Tile): boolean + sellProperty(property_id unsigned int): boolean + printProperties() + getPosition(): const unsigned int + changePosition(IN how_much unsigned int) + goJail() + checkJail() + isDead(): const boolean + plusGare() + minusGare() + getGareCount(): const unsigned int + checkBot(): const bool + killMe() + findToSell(IN rent int) + testRegPlayer() 	<ul style="list-style-type: none"> - wallet: int - collection: vector<link on Tile> + constructor, destructor + getBalance(): const int + changeBalance(IN amount int): bool + getNetWorth(): const int + getProperty(IN property_id): const like on Tile + addProperty(IN link on Tile) + removeProperty(IN property_id unsigned int): bool + getProperties(): const std::vector<of link on Tile> + collectionEmpty(): boolean + operatore << (IN ostream, IN inv Inventory) + printInventory(IN gare_count int) + killMe() + findPropertyWroth(IN price unsigned int): int + testRegInventory() 	<ul style="list-style-type: none"> - id: unsigned int - name: string - color: string - background: string - price: unsigned int - sell_price: unsigned int - owner: int + Constructor / Destructor + getType() + getId(): const int + getName(): const string + getColor(): const string + getBackColor(): const string + getPrice(): const unsigned int + getSellPrice(): const unsigned int + getOwner(): const int + sold() + bought(IN id_buyer unsigned int) + operator << (INOUT flux, IN Tile): flux

Mode texte

```
*****  
**      Bienvenue dans le jeu UNIVOPOLY      **  
***  
*****  
*****  
*****  
*****  
*****  
  
Salut, on joue à combien ?: 4  
#####  
  
C'est au tour du joueur 1!  
  
Joueur 1 lance les dés...  
Vous avez fait un 2 et un 4  
Vous vous déplacez à la case 6  
  
Case numero: 6  
Nom: Gouy  
La propriété est achetable vous pevez l'acheter pour 100€  
  
Tu as: 1000€  
Voudrai-tu l'acheter ? (y/n): y  
Joueur 1 à acheter Gouy  
  
Ce que tu possede:  
  
Tu as: 900€  
  
----- Nom: Gouy -----  
Id de la propriete: 6  
Prix de vente: 50€  
-----  
Passer le tour. (y/n):
```

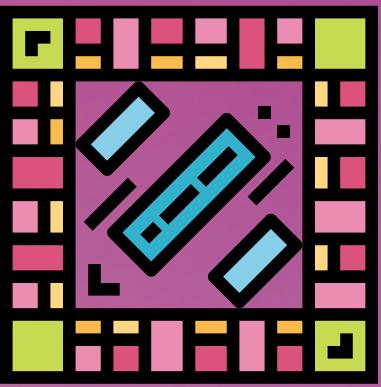
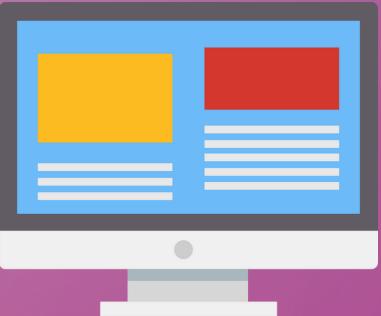
mainTxt

GameTxt

txtInit(nb_player)

txtLoop(game)

Partie graphique



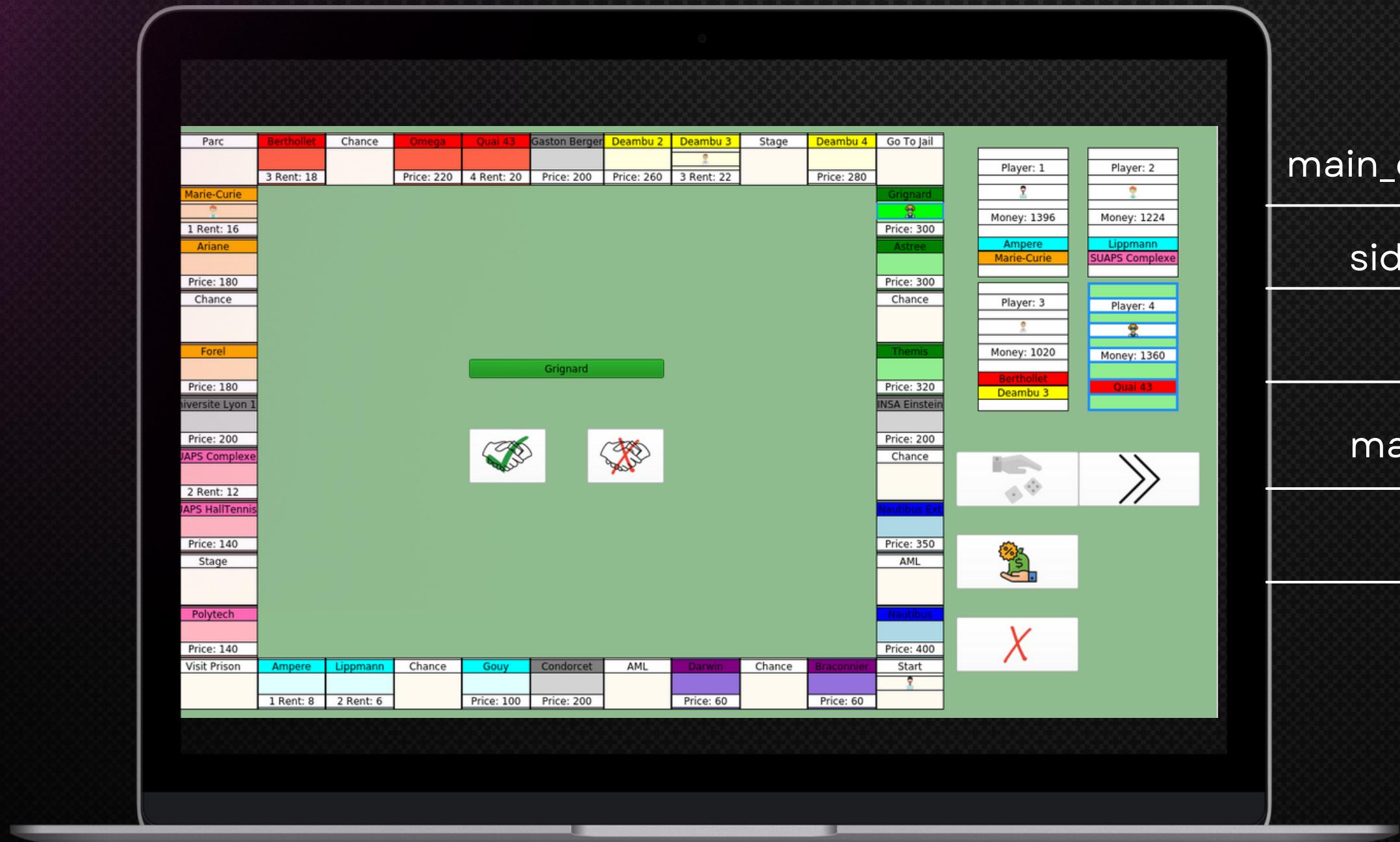
```
mainWindow
+ layout: link on QLayout
- sidebar: link on sideBar
- mainview: link on mainView
- game: link on Game
- current_player_index: integer

+ Constructor

signals:
+ diceRolled(INOUT amount integer)
+ playerMoved(vector of link on Player, integer)
+ askBuy(integer)
+ bought(vector of link on Player, integer)
+ tileStart(vector of link on Player, integer)
+ playersDisplayChange(vector of link on Player, integer)
+ boardDisplayChange(vector of link on Player, integer)
+ signalSellMenu(link on Player)
+ sold(vector of link on Player, integer)
+ playerDead()

slots:
- rollingDice()
- movingPlayer(IN amount)
- passingTurn()
- buying()
- slotSellMenu()
- selling(integer)
- tweaking()
```

Mode graphique



main_qt

sideBar

Controllers / Player_qt / Players

mainView

Board_qt / Tile_qt

MERCI POUR VOTRE ATTENTION

DES QUESTIONS ?

