2048

Contexte : Lors de ma découverte de PyQt6 pour applivisiteur (pour le contexte GSB), lors du tuto on nous montre une grille en 3x3 et j'ai eu un déclic, j'ai tout de suite pensé à faire un jeu du 2048 pour découvrir PyQt6.

J'ai commencé par importer toutes les fonctionnalités dont j'aurais besoin

```
from __future__ import annotations

import random
import sys
from typing import List

from PyQt6.QtCore import Qt    Impossible de résoudre l'importation « PyQt6.QtCore »
from PyQt6.QtGui import QFont, QKeyEvent    Impossible de résoudre l'importation « PyQt6.QtGui »
from PyQt6.QtWidgets import (    Impossible de résoudre l'importation « PyQt6.QtWidgets »
    QApplication,
    QGridLayout,
    QHBoxLayout,
    QLabel,
    QMainWindow,
    QPushButton,
    QVBoxLayout,
    QWidget,
)
```

J'ai défini les couleurs des cases et textes

```
SIZE = 4
TILE_COLORS = {
    0: "#cdc1b4",
    2: "#eee4da",
    4: "#ede0c8",
    8: "#f2b179",
    16: "#f59563",
    32: "#f67c5f",
    64: "#f65e3b",
    128: "#edcf72",
    256: "#edcc61",
    512: "#edc850",
    1024: "#edc53f",
    2048: "#edc22e",
}
TEXT_COLORS = {
    0: "#776e65",
    2: "#776e65",
    4: "#776e65",
    8: "#f9f6f2",
    16: "#f9f6f2",
    32: "#f9f6f2",
    64: "#f9f6f2",
    128: "#f9f6f2",
    256: "#f9f6f2",
    512: "#f9f6f2",
    1024: "#f9f6f2",
    2048: "#f9f6f2",
}
```

```python
class GameWidget(QWidget):
    def __init__(self) -> None:
        super().__init__()
        self.grid: List[List[int]] = [[0] * SIZE for _ in range(SIZE)]
        self.score = 0
        self.cells: List[List[QLabel]] = []
        self.setup_ui()
        self.new_game()

    def setup_ui(self) -> None:
        self.setWindowTitle("2048 - PyQt6")
        main_layout = QVBoxLayout()
        main_layout.setContentsMargins(10, 10, 10, 10)
        main_layout.setSpacing(10)

        header_layout = QHBoxLayout()
        self.score_label = QLabel("Score: 0")
        self.score_label.setFont(QFont("Arial", 16, QFont.Weight.Bold))
        header_layout.addWidget(self.score_label)

        self.restart_button = QPushButton("Recommencer")
        self.restart_button.setFont(QFont("Arial", 12, QFont.Weight.Bold))
        self.restart_button.clicked.connect(self.new_game)
        header_layout.addWidget(self.restart_button)

        main_layout.addLayout(header_layout)

        board_widget = QWidget()
        board_widget.setStyleSheet("background-color: #bbada0; border-radius: 10px;")
        board_layout = QGridLayout(board_widget)
        board_layout.setSpacing(10)
        board_layout.setContentsMargins(10, 10, 10, 10)

        for row in range(SIZE):
            row_cells: List[QLabel] = []
            for col in range(SIZE):
                label = QLabel()
                label.setAlignment(Qt.AlignmentFlag.AlignCenter)
                label.setFont(QFont("Arial", 24, QFont.Weight.Bold))
                label.setFixedSize(100, 100)
                label.setStyleSheet(self.tile_style(0))
                row_cells.append(label)
                board_layout.addWidget(label, row, col)
            self.cells.append(row_cells)

        main_layout.addWidget(board_widget)
        self.setLayout(main_layout)
```

Toute cette partie du code sert de base au jeu elle créer toute l'interface graphique.

```python
def new_game(self) -> None:
    self.grid = [[0] * SIZE for _ in range(SIZE)]
    self.score = 0
    self.spawn_tile()
    self.spawn_tile()
    self.update_ui()
```

Ici on remet tout à l'état originel c'est le bouton recommencer.

```python
def spawn_tile(self) -> None:
    empty_cells = [(r, c) for r in range(SIZE) for c in range(SIZE) if self.grid[r][c] == 0]
    if not empty_cells:
        return
    row, col = random.choice(empty_cells)
    self.grid[row][col] = 4 if random.random() < 0.1 else 2

def update_ui(self) -> None:
    self.score_label.setText(f"Score: {self.score}")
    for row in range(SIZE):
        for col in range(SIZE):
            value = self.grid[row][col]
            label = self.cells[row][col]
            label.setText(str(value) if value != 0 else "")
            label.setStyleSheet(self.tile_style(value))
            label.setProperty("value", value)
            label.style().unpolish(label)
            label.style().polish(label)
```

Ici on fais apparaitre les 'tuiles' (les cases) et on synchronise données du jeu avec affichage.

```python
def keyPressEvent(self, event: QKeyEvent) -> None:
    key = event.key()
    moved = False
    if key in (Qt.Key.Key_Left, Qt.Key.Key_Q):
        moved = self.move_left()
    elif key in (Qt.Key.Key_Right, Qt.Key.Key_D):
        moved = self.move_right()
    elif key in (Qt.Key.Key_Up, Qt.Key.Key_Z):
        moved = self.move_up()
    elif key in (Qt.Key.Key_Down, Qt.Key.Key_S):
        moved = self.move_down()

    if moved:
        self.spawn_tile()
        self.update_ui()
        if not self.can_move():
            self.game_over()
    else:
        super().keyPressEvent(event)
```

Ensuite je créer le mouvement quand on appuie sur les touches ZQSD (la base du gameplay)

Plus en détail :

```python
def move_left(self) -> bool:
    moved = False
    for i in range(SIZE):
        original = self.grid[i][:]
        row = self.compress(original)
        row = self.merge(row)
        row = self.compress(row)
        self.grid[i] = row
        if row != original:
            moved = True
    return moved

def move_right(self) -> bool:
    moved = False
    for i in range(SIZE):
        original = self.grid[i][:]
        reversed_row = list(reversed(original))
        row = self.compress(reversed_row)
        row = self.merge(row)
        row = self.compress(row)
        row = list(reversed(row))
        self.grid[i] = row
        if row != original:
            moved = True
    return moved
```

```python
def move_up(self) -> bool:
    moved = False
    for col in range(SIZE):
        column = [self.grid[row][col] for row in range(SIZE)]
        original = column[:]
        column = self.compress(column)
        column = self.merge(column)
        column = self.compress(column)
        for row in range(SIZE):
            self.grid[row][col] = column[row]
        if column != original:
            moved = True
    return moved

def move_down(self) -> bool:
    moved = False
    for col in range(SIZE):
        column = [self.grid[row][col] for row in range(SIZE)]
        original = column[:]
        column.reverse()
        column = self.compress(column)
        column = self.merge(column)
        column = self.compress(column)
        column.reverse()
        for row in range(SIZE):
            self.grid[row][col] = column[row]
        if column != original:
            moved = True
    return moved
```

Enfin la possibilité si la case peut bouger et donc le game over si aucune ne peut

```
    def can_move(self) -> bool:
        for row in range(SIZE):
            for col in range(SIZE):
                if self.grid[row][col] == 0:
                    return True
                if col + 1 < SIZE and self.grid[row][col] == self.grid[row][col + 1]:
                    return True
                if row + 1 < SIZE and self.grid[row][col] == self.grid[row + 1][col]:
                    return True
        return False

    def game_over(self) -> None:
        self.score_label.setText(f"Perdu ! Score final: {self.score}")
```

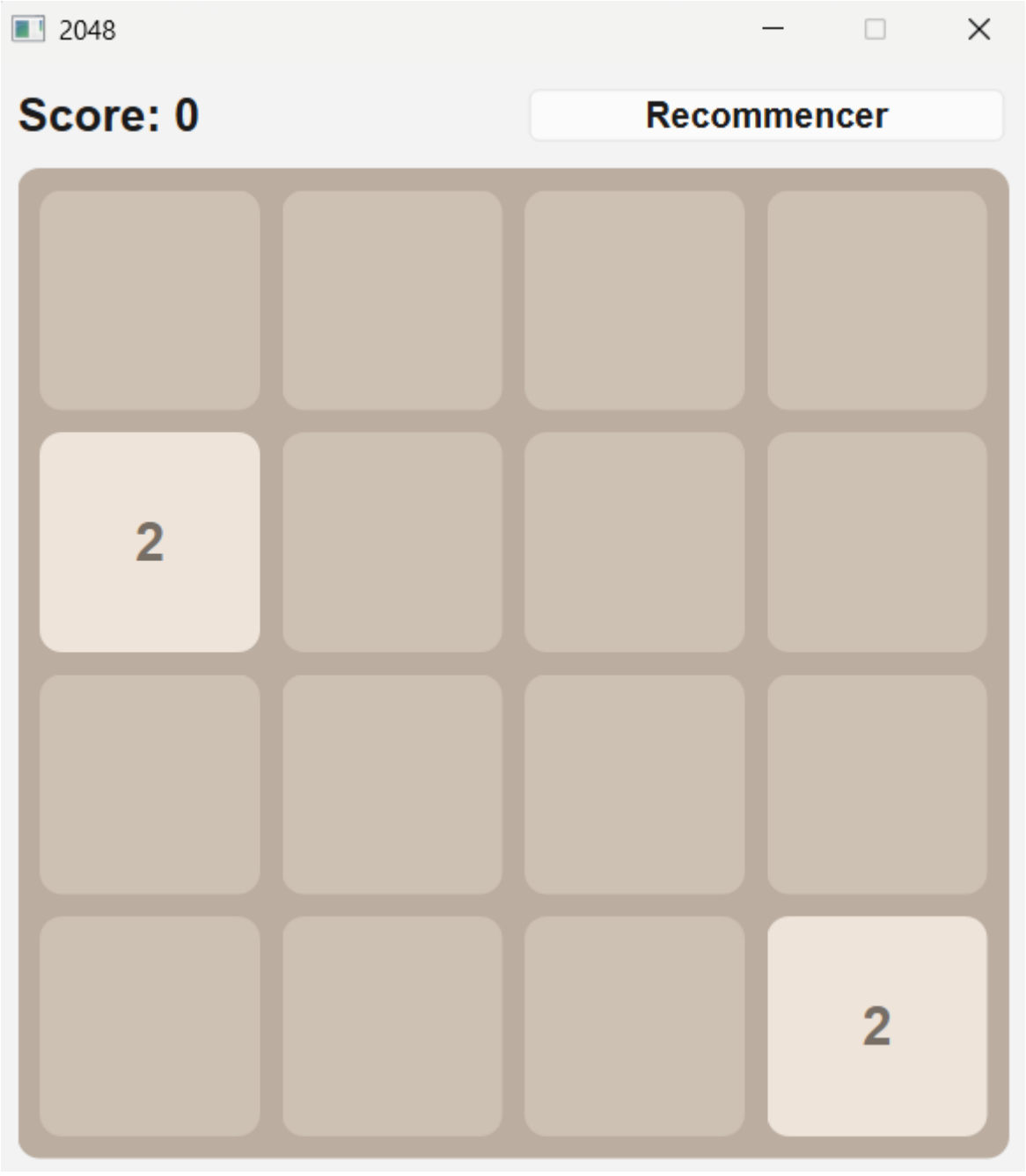
```
class MainWindow(QMainWindow):
    def __init__(self) -> None:
        super().__init__()
        self.setWindowTitle("2048")
        self.game_widget = GameWidget()
        self.setCentralWidget(self.game_widget)
        self.setFixedSize(self.sizeHint())

    def keyPressEvent(self, event: QKeyEvent) -> None:
        self.game_widget.keyPressEvent(event)


def main() -> None:
    app = QApplication(sys.argv)
    window = MainWindow()
    window.show()
    sys.exit(app.exec())


if __name__ == "__main__":
    main()
```

Pour finir la fenêtre de jeu qui apparait quand on lance l'appli.

Afin de ne plus passer par VSC pour jouer au jeu je le créer en .exe pour cela je lance

pip install pyinstaller
Puis py -m PyInstaller --onefile --windowed 2048.py

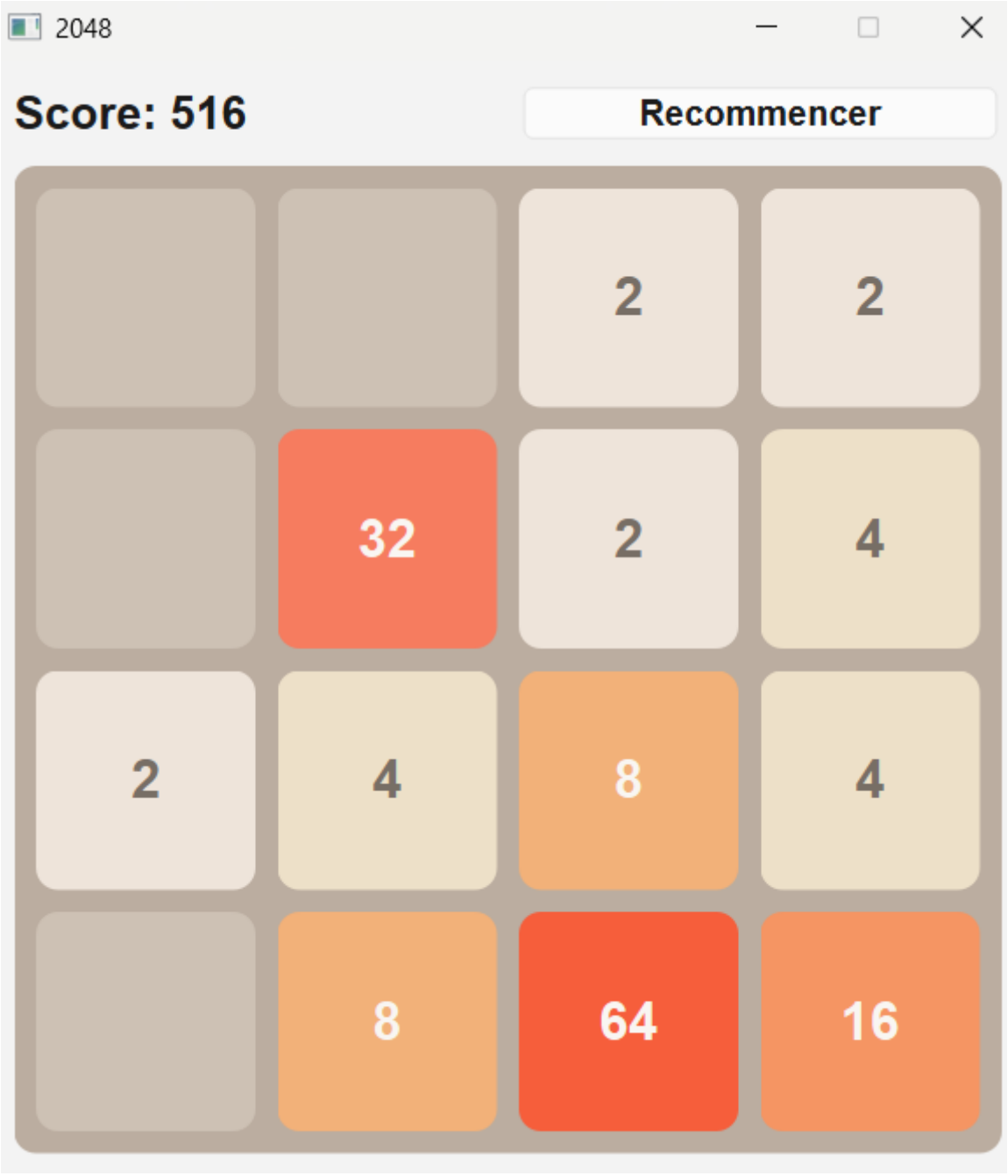
Voilà 2048.exe créer dans dist/ je peux changer l'icon via Prorpiété après un clic droit comme n'importe quel app windows. Je peux envoyer ce .exe a tout user sous windows 10/11 pour qu'il y joue.



Voilà le jeu au lancement

Et après quelques coups joués :

2048



COUR Cyrille – cyrillecour11@gmail.com – Aristée 2025/2027