

## Présentation

Le jeu **2048** se joue sur une grille de 4×4 cases, avec des tuiles de valeurs variées (mais toujours des puissances de deux) pouvant être déplacées dans les quatre directions (**G**auche, **D**roite, **H**aut et **B**as).

Lors d'un mouvement, l'ensemble des tuiles du plateau sont déplacées dans la même direction jusqu'à rencontrer les bords du plateau ou une autre tuile sur leur chemin. Si deux tuiles de même valeur entrent en collision durant le mouvement, elles fusionnent en une nouvelle tuile de valeur double.

Après chaque mouvement, une tuile portant un **2** ou un **4** apparaît dans une case vide de manière aléatoire.

2048				x
1024	512	256	256	
4	8	32	8	
	4	2	2	
2		4		

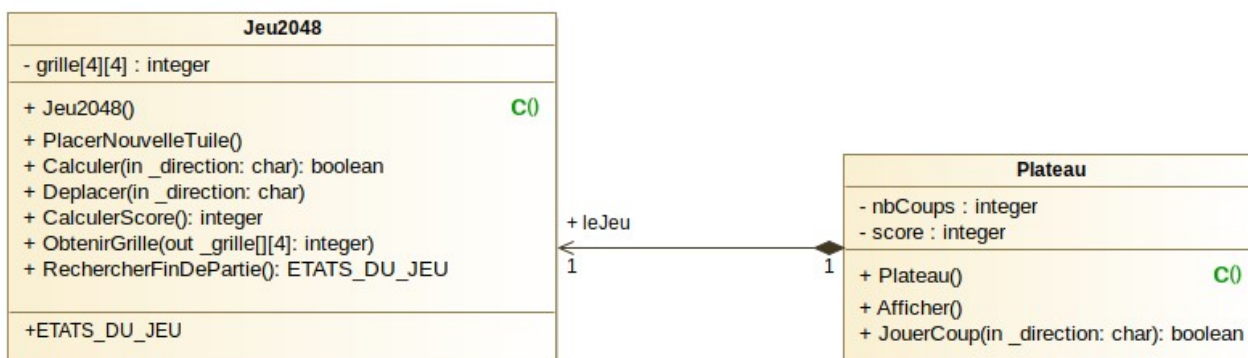
*Un exemple de fonctionnement est proposé en annexe.*

La partie est gagnée lorsque la valeur 2048 apparaît sur le plateau. Elle est perdue si toutes les cases sont occupées et que plus aucun mouvement n'est possible.

## Diagramme de classes de l'application

Comme bien souvent pour plus de portabilité, on cherche à séparer l'affichage des données et leur traitement. C'est le cas ici, l'application est divisée en deux classes :

- La classe **Plateau** chargée de l'affichage et l'enchaînement des opérations.
- La classe **Jeu2048**, elle réalise le traitement des données sur la grille

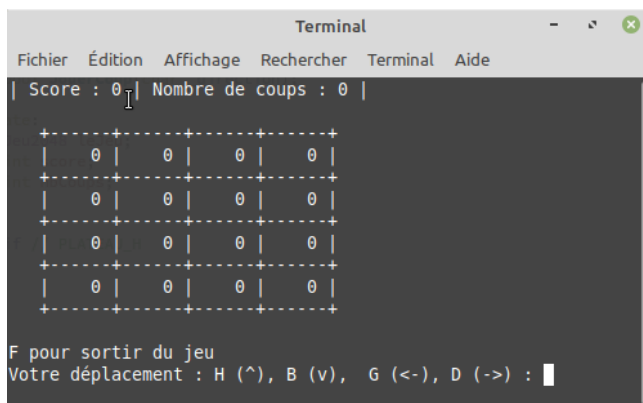


Dans la classe **Jeu2048**, **ETATS\_DU\_JEU** représente les différents états que peut prendre ce jeu : **EN\_COURS**, **PERDU** et **GAGNE**.

*Vous êtes invité à respecter les différentes étapes  
pour être en mesure de jouer avec le 2048 que vous aurez codé.*

## Travail à réaliser

1. Avec votre environnement de développement, créez un nouveau projet **C++** en mode application **console**, nommé **LeJeuDu2048**.
2. Après avoir ajouté un fichier source **main.cpp** ajouter les deux classes du projet en respectant le diagramme de classes proposé.
3. Implémentez la relation de composition en tant qu'attribut automatique dans la classe **Plateau**, l'allocation dynamique n'est pas nécessaire ici.
4. Codez le constructeur de la classe **Jeu2048**. Dans un premier temps, il remplit la grille de 0, la valeur indiquant que la case est vide.
5. Codez la méthode `void Jeu2048::ObtenirGrille(int _grille[][TAILLE])` attention, le paramètre est en sortie, sa déclaration ne doit pas être précédée du mot clé **const**. Le tableau contenu dans la classe **Jeu2048** est recopié dans celui passé paramètre.
6. Codez le constructeur de la classe **Plateau**. Son seul rôle est d'initialiser les deux attributs avec la valeur 0. Le score est nul au départ ainsi que le nombre de coups joués.
7. Complétez la méthode `void Plateau::Afficher()` comme le montre l'exemple ci-dessous, elle réalise l'affichage du plateau. Dans un premier temps, les 0 seront affichés.



L'affichage utilise le flux de sortie **cout**.

Pour maintenir la largeur de colonne identique, le manipulateur **setw()** permet de définir une largeur de l'affichage.

Éventuellement le manipulateur **setfill()** permet de remplir avec un caractère particulier l'espace défini par **setw()**.

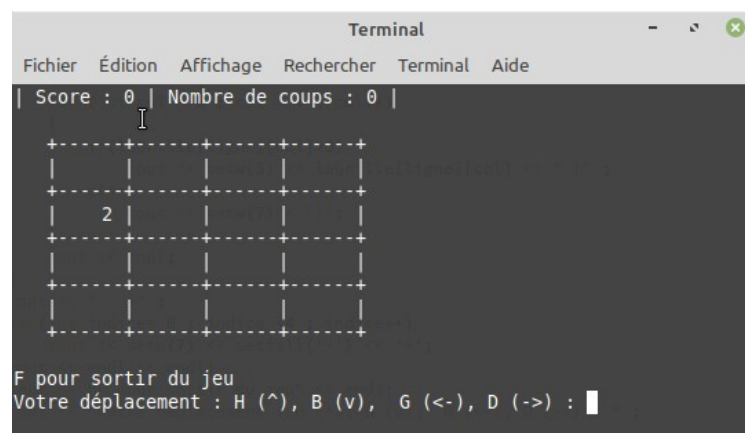
```
void Plateau::Afficher()
{
    int laGrille[TAILLE][TAILLE];
    system("clear");
    leJeu.ObtenirGrille(laGrille);
    cout << "| Score : " << score << " | Nombre de coups : " << nbCoups << " |" << endl;
    cout << endl;
    for(int ligne = 0 ; ligne < 4 ; ligne++)
    {
        cout << "  + " ;
        for(int indice= 0 ; indice <4 ; indice++)
            cout << setw(7) << setfill('-') << '+';

        // affichage d'une ligne de la grille
    }
    // affichage de la dernière ligne du tableau
    // affichage du texte sous le tableau
}
```

8. Codez la méthode `void Jeu2048::PlacerNouvelleTuile()`. Son rôle est de placer dans une case non occupée, choisie au hasard, une nouvelle tuile possédant la valeur 2 ou 4. On considère qu'il y a au moins une case de libre sur la grille. La fonction doit boucler jusqu'à l'obtention d'une case libre et la valeur aléatoire déposée.

Pour rappel, `rand() % 4` donne une valeur entre 0 et 3 et `((rand() % 2) + 1) * 2` donne soit la valeur 2 soit la valeur 4. Pour que le tirage soit le plus au hasard possible, il est nécessaire d'ajouter dans le constructeur de la classe les instructions : `srand(time(nullptr))`; Les fonctions de hasard se trouvent dans la librairie `math.h` et la gestion de l'heure dans `time.h`.

9. Ajoutez également au constructeur de la classe **Jeu2048** l'appel de la méthode **PlacerNouvelleTuile()**. Puis modifier la méthode **Afficher()** de la classe **Plateau** pour qu'elle affiche les valeurs uniquement lorsque la case n'est pas vide, différente de 0.



10. Complétez la méthode **Deplacer()** de la classe **Jeu2048** afin de déplacer les différentes valeurs en fonction de la direction choisie par l'utilisateur.

```
void Jeu2048::Deplacer(char _direction)
{
    int ligne;
    int colonne;
    for(int indice = 0 ; indice < 4 ; indice++)
    {
        switch (_direction)
        {
            case 'G':
                for(ligne = 0 ; ligne < 4 ; ligne++)
                {
                    for(colonne = 0 ; colonne < 3 ; colonne++)
                    {
                        if(grille[ligne][colonne] == 0)
                        {
                            grille[ligne][colonne] = grille[ligne][colonne+1];
                            grille[ligne][colonne+1] = 0;
                        }
                    }
                }
                break;
                // à poursuivre pour les autres cas
        }
    }
}
```

11. Vous allez maintenant commencer le codage de la méthode **JouerCoup()** comme le montre l'exemple ci-dessous afin de vérifier les déplacements dans la grille. Complétez votre programme principal pour qu'il appelle cette méthode jusqu'à ce que l'utilisateur appuie sur la touche 'F'.

```
bool Plateau::JouerCoup()
{
    bool retour = false;
    char touche;
    cin >> touche ;
    touche = toupper(touche);
    if(strchr("BHGD",touche) != nullptr)
    {
        nbCoups++;
        leJeu.Deplacer(touche);
        leJeu.PlacerNouvelleTuille();
        Afficher();
    }
    if(touche == 'F')
        retour = true;
    return retour;
}
```

12. En vous inspirant de la méthode **Deplacer()** codez la méthode **Calculer()** de la classe **Jeu2048**. En fonction de la direction, elle ajoute le contenu de deux cases adjacentes possédant la même valeur, sans oublier de remplacer la première valeur par 0, une case se libère après le calcul. Elle retourne vrai si au moins un calcul a été effectué, faux sinon.
13. Complétez la méthode **JouerCoup()** afin d'effectuer le calcul après avoir déplacé les cases de la grille. Attention, si un calcul a été effectué, il faut penser à bouger à nouveau les cases dans la même direction, il y a des cases vides qui peuvent être apparues lors du précédent calcul.
14. Codez la méthode **CalculezScore()**. Cette méthode effectue la somme des cases de la grille. Réalisez l'appel de cette méthode dans la méthode **JouerCoup()** après avoir effectué tous les déplacements.
15. Codez la méthode **RechercherFinDePartie()** de la classe **Jeu2048** en respectant les règles énoncées lors de la présentation. Si la partie n'est pas terminée, la valeur de retour est **EN\_COURS**.
16. Terminez le codage de la méthode **JouerCoup()** afin de placer une nouvelle tuile lorsque le jeu n'est pas terminé ou pour les autres situations un message indiquant que la partie est gagnée ou perdue.

## Annexe

Voici une version partielle d'une session de jeu où successivement on peut découvrir la situation actuelle du jeu, une invite à proposer un nouveau déplacement. La réponse formulée par le joueur en un seul caractère.

---

En rouge, la nouvelle tuile tirée au hasard.

---

```
-----  
| | | 4 |  
-----  
| | |2 |8 |  
-----  
| |2 |8 |16|  
|2 |4 |32|64|  
-----
```

← Grille à un instant donné

Déplacement demandé pour  
la situation suivante.

Votre déplacement : H (haut), B (bas), G (gauche), D (droite)  
Après un déplacement sur la gauche, la grille devient :

```
-----  
|4 | | |2 |  
-----  
|2 |8 | | |  
-----  
|2 |8 |16| |  
-----  
|2 |4 |32|64|  
-----
```

← Une nouvelle tuile apparaît

Votre déplacement : H (haut), B (bas), G (gauche), D (droite)

```
-----  
| | |2 | |  
-----  
|4 | | | |  
-----  
|2 |16|16|2 |  
-----  
|4 |4 |32|64|  
-----
```

Votre déplacement : H (haut), B (bas), G (gauche), D (droite)

```
-----  
| | |2 |2 |  
-----  
| | | |4 |  
-----  
| |2 |32|2 |  
-----  
| |8 |32|64|  
-----
```

Votre déplacement : H (haut), B (bas), G (gauche), D (droite)

```
-----  
| | |2 | |  
-----  
| | |4 |4 |  
-----  
| |2 |2 |2 |  
-----  
| |8 |64|64|  
-----
```