

# PROJET SUIKA-GAME



Ba Cheikh  
Bah Mamadou  
Yrius Marc

# PRINCIPE DU JEU







# MENU SDL

```
void MenuSDL::boucleMenu()
{
    int x,y;
    SDL_Event event;
    do
    {
        while(SDL_PollEvent(&event))
        {
            switch(event.type)
            {
                case SDL_QUIT:
                    continu = SDL_FALSE;
                    break;
                case SDL_MOUSEBUTTONDOWN:
                    x = event.motion.x;
                    y = event.motion.y;
                    bool fichier = fichierVide("../data/save.txt");
                    if ((x >= 120 && x <= 600)
                        && (y >= 263 && y <= 333))
                    {
                        fermerMenu();
                        JeuSDL j;
                        j.boucleDeJeu();
                    }

                    if ((x >= 120 && x <= 600)
                        && (y >= 366 && y <= 433) && !fichier)
                    {
                        fermerMenu();
                        JeuSDL j;
                        j.getPartie().recuperer();
                        j.boucleDeJeu();
                    }

                    break;
            }
        }
    }
    afficher();
    SDL_RenderPresent(renderer);
}
```

```
bool fichier_vide = fichierVide("../data/save.txt");

if(fichier_vide)
{
    cont.loadFromFile("../data/menu/CONTINUE_f.png",renderer,2);
}
else
{
    cont.loadFromFile("../data/menu/CONTINUE.png",renderer,2);
}

cont.setPos(100,350,TAILLE_SPRITE, TAILLE_SPRITE);

score.loadFromFile("../data/menu/score.png",renderer,2);
score.setPos(10,10,TAILLE_SPRITE, TAILLE_SPRITE);

font = TTF_OpenFont("../data/utile/arial.ttf",25);
if(font==NULL) TTF_CloseFont(font);

score1.setPos(85,49,30,30);
score2.setPos(85,121,30,30);
score3.setPos(85,190,30,30);

boucleMenu();
```

# CLASSE GESTION DU JEU

```
void Jeu::sauvegarder()
{
    string filename = "../data/save.txt";
    ofstream fichier(filename.c_str());
    assert(fichier.is_open());
    fichier << score << endl;
    fichier << fruit_actuel.getIdFruit() << endl;
    fichier << fruit_suivant.getIdFruit() << endl;
    fichier << tableau_fruit.size() << endl;
    for (int i = 0; i < (int)tableau_fruit.size(); i++)
    {
        fichier << tableau_fruit[i].getIdFruit() << endl;
        fichier << tableau_fruit[i].getPosition().x << ' ' << tableau_fruit[i].getPosition().y << endl;
    }
}

void Jeu::recuperer()
{
    string filename = "../data/save.txt";
    ifstream fichier(filename.c_str());
    int fa, fs, size;
    fichier >> score >> fa >> fs >> size;
    fruit_actuel.setFruit(fa);
    fruit_suivant.setFruit(fs);

    int Id;
    float x, y;
    for (int i = 0; i < size; i++)
    {
        fichier >> Id >> x >> y;
        tableau_fruit[i].setFruit(Id);
        tableau_fruit[i].setPosition(x, y);
    }
}
```

```

void Jeu::fusionner_fruit()
{
    // Créer un nouveau tableau pour stocker les fruits fusionnés
    vector<Fruit> nouveaux_fruits;

    // Parcourir le tableau de fruits
    for (int i = 0; i < (int)tableau_fruit.size(); ++i)
    {
        // Parcourir à partir du fruit suivant
        for (int j = i + 1; j < (int)tableau_fruit.size(); ++j)
        {
            // Vérifier si les fruits ont le même identifiant et si cet identifiant est inférieur à 11
            if (tableau_fruit[i].getIdFruit() == tableau_fruit[j].getIdFruit() && tableau_fruit[i].getIdFruit() < 11)
            {
                // Calculer la position moyenne des deux fruits
                Vec2 temp = (tableau_fruit[i].getPosition() + tableau_fruit[j].getPosition()) / 2.0;

                // Créer un nouveau fruit avec l'identifiant suivant
                Fruit nouveau_fruit = genereFruit(tableau_fruit[i].getIdFruit() + 1);

                // Définir la position du nouveau fruit
                nouveau_fruit.setPosition(round(temp.x), round(temp.y));

                // Ajouter le nouveau fruit au tableau des nouveaux fruits
                nouveaux_fruits.push_back(nouveau_fruit);

                // Incrémenter le score
                increment_score(tableau_fruit[i].getIdFruit() * tableau_fruit[j].getIdFruit());

                // Sortir de la boucle intérieure après avoir trouvé une paire de fruits à fusionner
                break;
            }
        }
    }

    // Effacer le tableau de fruits actuel
    tableau_fruit.clear();

    // Ajouter les nouveaux fruits au tableau de fruits
    for (int i = 0; i < (int)nouveaux_fruits.size(); ++i)
    {
        tableau_fruit.push_back(nouveaux_fruits[i]);
    }
}

```

```

void Jeu::physique_fruit()
{
    for (int i = 0; i < (int)tableau_fruit.size() - 1; ++i)
    {
        maj_vitesse_position_fruit(tableau_fruit[i], 9.81);
        collision_Mur(tableau_fruit[i]);
        for (int j = 0; j < (int)tableau_fruit.size() - 1; ++j)
        {
            if (i != j)
            {
                collision_Entre_Fruit(tableau_fruit[i], tableau_fruit[j]);
            }
        }
    }
}

```

# JEU SDL & SPRITE

```
Sprite::Sprite(const Sprite &im)
{
    *this = im;
}

Sprite &Sprite::operator=(const Sprite &im)
{
    surface = im.surface;
    texture = im.texture;

    return *this;
}

void Sprite::drawScore(SDL_Renderer *renderer, TTF_Font * font, SDL_Color color, int score)
{
    char buffer[20] = {'\0'};

    citoa(score, buffer, 10);

    surface = TTF_RenderText_Solid(font, buffer, color);
    texture = SDL_CreateTextureFromSurface(renderer, surface);
    SDL_FreeSurface(surface);
    SDL_RenderCopy(renderer, texture, NULL, &pos);
}
```



```
void JeuSDL::initTabSprite()
{
    Plateau.loadFromFile("../data/utile/plateau_suika.png", renderer, 0);
    Plateau.setPos(157, 401, TAILLE_SPRITE, TAILLE_SPRITE);

    tabLoad[0].loadFromFile("../data/fruits/circle0.png", renderer, 1);
    tabLoad[1].loadFromFile("../data/fruits/circle1.png", renderer, 1);
    tabLoad[2].loadFromFile("../data/fruits/circle2.png", renderer, 1);
    tabLoad[3].loadFromFile("../data/fruits/circle3.png", renderer, 1);
    tabLoad[4].loadFromFile("../data/fruits/circle4.png", renderer, 1);
    tabLoad[5].loadFromFile("../data/fruits/circle5.png", renderer, 1);
    tabLoad[6].loadFromFile("../data/fruits/circle6.png", renderer, 1);
    tabLoad[7].loadFromFile("../data/fruits/circle7.png", renderer, 1);
    tabLoad[8].loadFromFile("../data/fruits/circle8.png", renderer, 1);
    tabLoad[9].loadFromFile("../data/fruits/circle9.png", renderer, 1);
    tabLoad[10].loadFromFile("../data/fruits/circle10.png", renderer, 1);
}

// fabrique tout ce qui est necessaire pour le fruit actuel
void JeuSDL::drawFruitActuel()
{
    fruitActuel.setPos(partie.getPositionFruitActuel().x, partie.getPositionFruitActuel().y
        ,TAILLE_SPRITE, TAILLE_SPRITE);
    SDL_RenderCopy(renderer, tabLoad[partie.getIdFruitActuel()-1].getTexture(), nullptr, &fruitActuel.getPosition());
}

void JeuSDL::drawFruitSuivant()
{
    fruitActuel.setPos(fruitSuivant.getPosition().x, fruitSuivant.getPosition().y
        ,TAILLE_SPRITE, TAILLE_SPRITE);
    SDL_RenderCopy(renderer, tabLoad[partie.getIdFruitSuivant()-1].getTexture(), nullptr, &fruitSuivant.getPosition());
}
```





CE QUE NOUS AVONS RETENU

CE QUE NOUS AVONS APPRIS

CE QUI NOUS A PLU DANS LE PROJET



MERCI DE VOTRE ATTENTION