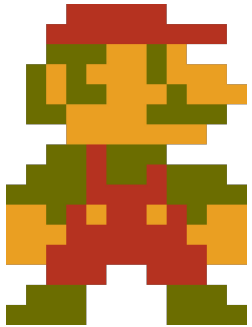
Pràctica PRO2



PART 1

Quadrimestre Primavera 2024/25

RogeR Bitlloch | *roger.bitlloch@estudiantat.upc.edu*

Part 1: Un nou tipus d'objecte

Es tracta de fer un nou tipus d'objecte que el `Mario` pugui recollir. El joc original tenia monedes, però pot ser qualsevol altra cosa.

Els requeriments d'aquest objecte són:

1. Que n'hi pugui haver molts en el joc, en llocs accessibles (o no tant!).
2. Que quan el `Mario` passa pel damunt els agafi implícitament i per tant, desapareguin de la pantalla.
3. Que es tingui un comptatge de quants objectes s'han agafat. Donat que el codi inicial no proporciona utilitats per mostrar text, no cal mostrar el comptatge per pantalla textualment.
4. Que tinguin una animació senzilla. Poden "flotar" en l'aire amb moviments verticals de tipus sinusoidal, o bé rotar (cosa que implica fer "sprites" diferents i anar-los mostrant consecutivament), o qualsevol altra cosa que atregui l'atenció del jugador.

> **ENTREGA 1**: Cal fer una classe nova pel nou tipus d'objecte, tal com s'ha explicat, i estructurar-la de tal manera que encaixi perfectament amb la resta del joc. Com ha passat amb els exercicis de la Part 0, es pot modificar la classe `Game` i afegir-hi dades noves o mètodes nous (tot i respectant la lògica actual), apart de fer la nova classe.

1.1.1 Crear classe i nou objecte

```
class Strawberry {
    private:
        pro2::Pt base_pos_;
        pro2::Pt pos_;
        bool collected_ = false;
```

```
// Constructora
Strawberry::Strawberry(Pt pos) {
    pos_ = pos;
    base_pos_ = pos;
}
```

Podeu crear un sprite a partir de la imatge que vulgueu utilitzant aquesta web:
image2cpp

(Abans caldrà que reescaleu la imatge al tamany “pixel art” amb alguna web rotllo Resize images)

(Després caldrà que poseu les claus a l’inici i final de cada columna)

1.1.2 Creeu funció paint

```
const int Width  = 18;  // ample de l'sprite
const int Height = 16;  // alçada de l'sprite

const vector<vector<vector<int>>> Strawberry::strawberry_sprites_ = {
    { // 'normal00', 18x16px
    {-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, -1, 0x006abe30, -1, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, 0x006abe30, -1, 0x0037946e, 0x006abe30, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, 0x006abe30, 0x00491675, 0x006abe30, 0x006abe30, 0x00491675, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, 0x00491675, 0x008a0f36, 0x00491675, 0x00491675, 0x008a0f36, 0x00491675, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, 0x008a0f36, 0x00de2a2a, 0x00de2a2a, 0x00de2a2a, 0x00de2a2a, 0x00ff5f42, 0x008a0f36, 0x008a0f36, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, 0x00ff5f42, 0x00de2a2a, 0x00de2a2a, 0x00de2a2a, 0x00ff5f42, 0x00de2a2a, 0x00de2a2a, 0x00de2a2a, 0x00ff5f42, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, 0x008a0f36, 0x00de2a2a, 0x00de2a2a, 0x008a0f36, 0x00de2a2a, 0x00ff5f42, 0x00de2a2a, 0x008a0f36, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, 0x00de2a2a, 0x008a0f36, 0x00de2a2a, 0x00de2a2a, 0x00de2a2a, 0x008a0f36, 0x008a0f36, 0x00de2a2a, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, 0x00de2a2a, 0x008a0f36, 0x008a0f36, 0x008a0f36, 0x008a0f36, 0x008a0f36, 0x00de2a2a, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, 0x00de2a2a, 0x008a0f36, 0x008a0f36, 0x00de2a2a, -1, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, 0x00de2a2a, 0x00de2a2a, -1, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
    },
    { // 'normal01', 18x16px
    {-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
    {-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
```

Allò que vulgueu que sigui transparent substituir-ho per un -1

```
void Strawberry::paint(pro2::Window& window) {
    const Pt top_left = {pos_.x - Width/2, pos_.y - Height - 1};
    paint_sprite(window, top_left, actual_sprite, false);
}
```

Funció paint per pintar l'sprite

1.1.3 Declareu-les a game perquè apareguin al taulell

```
class Game {
    std::vector<Strawberry> strawberrys_;
    int strawberry_count_ = 0;
```

```
Game::Game(int width, int height)
    : mario_({width / 2, 150}),
      platforms_{
          Platform(100, 300, 200, 211),
          Platform(0, 200, 250, 261),
          Platform(250, 400, 150, 161),
      },
      strawberrys_{
        Strawberry({200, 200}),
        Strawberry({100, 250}),
        Strawberry({325, 150}),
      },
      finished_(false) {
    for (int i = 1; i < 20; i++) {
        platforms_.push_back(Platform(250 + i * 200, 400 + i * 200, 150, 161));
        strawberrys_.push_back(Strawberry({325 + i * 200, 150}));
    }
}
```

1.2.1 Que quan el mario passi per damunt les agafi implícitament

```
bool Strawberry::is_collected() const {
    return collected_;
}

void Strawberry::collect() {
    collected_ = true;
}
```

Afegim al nostre col·leccionable l'atribut collected_ per indicar si s'ha de mostrar o no

```
void Strawberry::paint(pro2::Window& window) {
    if (collected_) return;

    const Pt top_left = {pos_.x - Width/2, pos_.y - Height - 1};
    paint_sprite(window, top_left, actual_sprite, false);
}

void Strawberry::update()
{
    if (collected_) return;
```

Retornem en el cas que collected_ sigui cert i, per tant, no volguem que es mostri per pantalla

1.2.2 Revisar si Mario toca una maduixa

```
void Game::update_objects(pro2::Window& window) {
    mario_.update(window, platforms_);

    for (Strawberry& s : strawberrys_) {
        s.update();

        if (!s.is_collected() && rects_overlap(mario_.get_rect(), s.get_rect())) {
            s.collect();
        }
    }
}
```

Per cada fotograma revisarem si el “collider” del Mario entra en col·lisió amb el “collider” de la maduixa.
get_rect(): retorna un rectangle amb les posicions que ocupa el personatge
rects_overlap(Rect r1, Rect r2): retorna cert si algun píxel dels rectangles coincideix

1.3.1 Comptatge de quants objectes has agafat

```
void Game::update_objects(pro2::Window& window) {
    mario_.update(window, platforms_);

    for (Strawberry& s : strawberrys_) {
        s.update();

        if (!s.is_collected() && rects_overlap(mario_.get_rect(), s.get_rect())) {
            s.collect();
            ++strawberry_count_;
            cout << "Maduixa recollida! Tens: " << strawberry_count_ << " maduixes!!!" << endl;
        }
    }
}
```

Inicialitzem a 0 una variable comptador i cada cop que agafem un col·leccionable l'augmentem i ho imprimim per pantalla

1.4.1 Animació amunt i avall sinusoidalment

```
class Strawberry {
    private:
        pro2::Pt base_pos_;
        pro2::Pt pos_;
        bool collected_ = false;

        float    phase_   = 0.f;   // φ, permet desfasar maduixes veïnes
        float kAmp     = 4.f;      // A (píxels)
        int kPeriodFrames  = 90;     // T (fotogrames)
        float kOmegaFrame   = 2 * M_PI / kPeriodFrames;   // ω
```

```
void Strawberry::update()
{
    if (collected_) return;

    phase_ += kOmegaFrame;          // avançar el temps
    if (phase_ > 2 * M_PI) phase_ -= 2 * M_PI;

    pos_.y = base_pos_.y - kAmp + kAmp * sin(phase_);
}
```

(Li demaneu al chatGpt i us ho fa, però bàsicament és la fórmula de l'ona sinusoidal)

pos_.y(t) = base_pos_.y + A*sin(wt)

1.4.2 Animació varis sprites

Crear vector de sprites, i anar-los pintant segons el frame actual, podeu triar ja vosaltres cada quants frames poseu el següent (jo ho faig cada 4)

```
const vector<vector<vector<int>>> Strawberry::strawberry_sprites_ = {···

void Strawberry::paint(pro2::Window& window) {
    if (collected_) return;

    const Pt top_left = {pos_.x - Width/2, pos_.y - Height - 1};
    vector<vector<int>> actual_sprite = strawberry_sprites_[(window.frame_count()%(strawberry_sprites_.size()*4))/4];
    paint_sprite(window, top_left, actual_sprite, false);
}
```