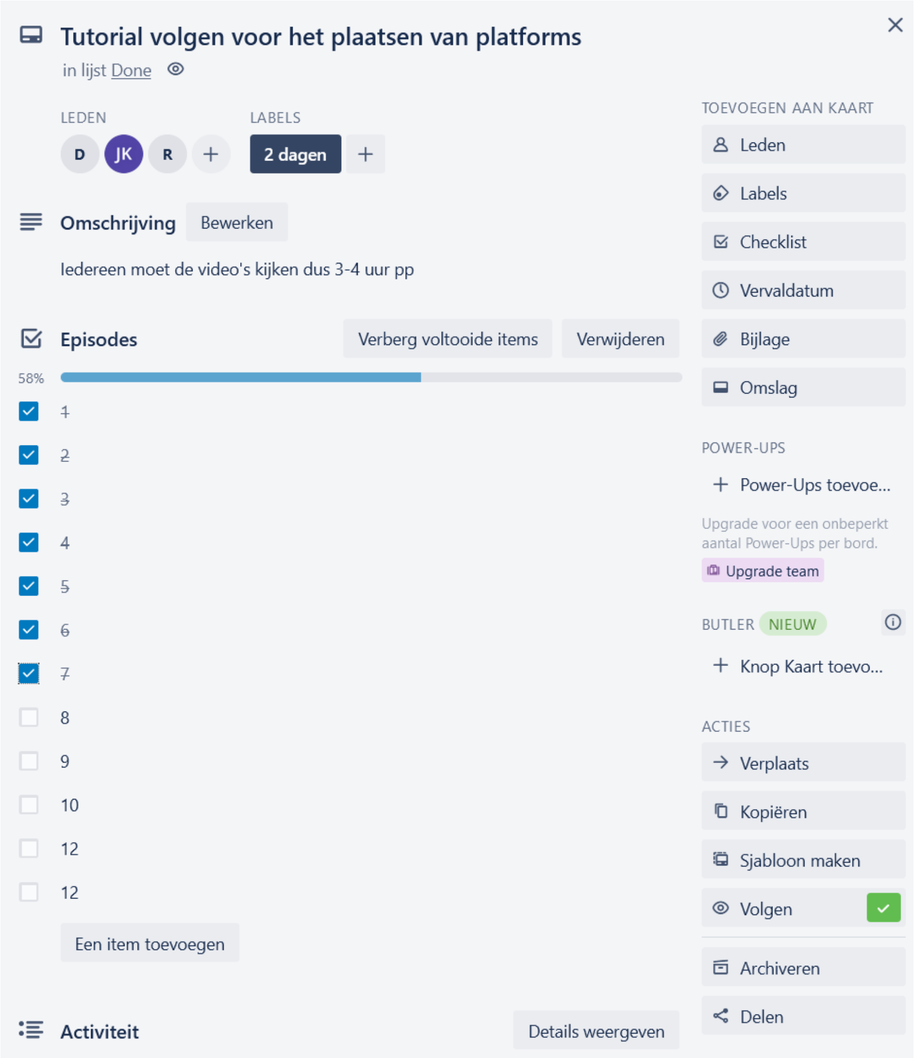
Code User Story
Roel

Tutorial gevolgd en juiste gravity gemaakt



Tutorial volgen voor het plaatsen van platforms

in lijst Done

LEDEN: D JK R +

LABELS: 2 dagen +

Omschrijving Bewerken

Iedereen moet de video's kijken dus 3-4 uur pp

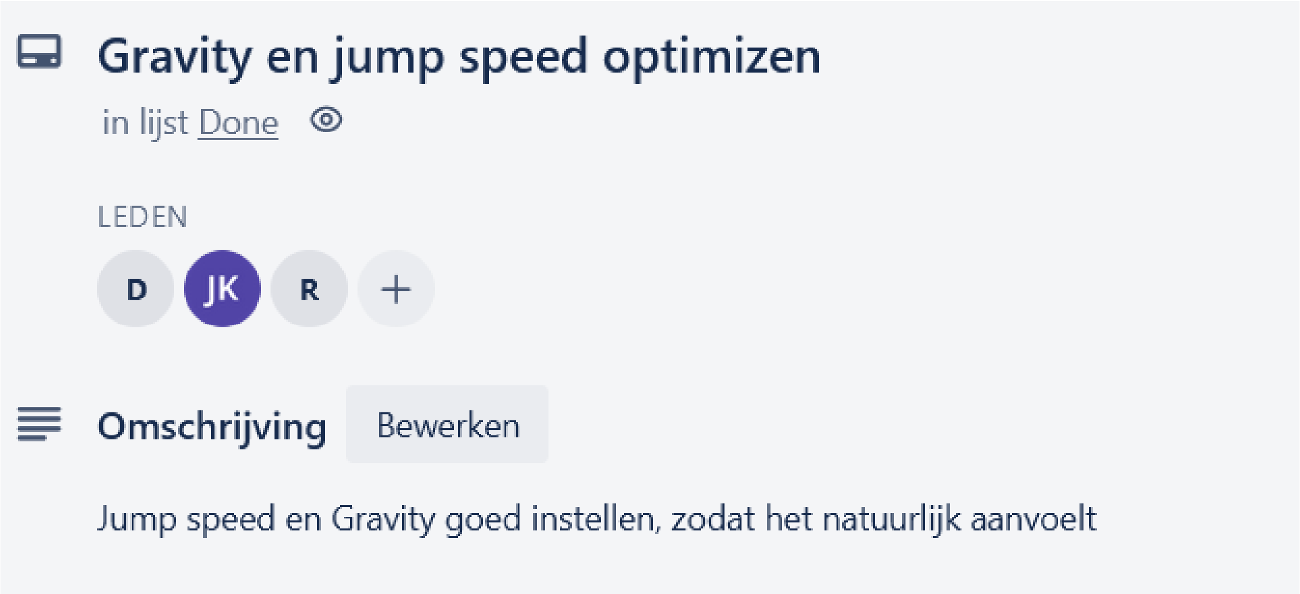
Episodes Verberg voltooide items Verwijderen

58%

- ☑ ~~1~~
- ☑ ~~2~~
- ☑ ~~3~~
- ☑ ~~4~~
- ☑ ~~5~~
- ☑ ~~6~~
- ☑ ~~7~~
- □ 8
- □ 9
- □ 10
- □ 12
- □ 12

Een item toevoegen

Activiteit Details weergeven



Gravity en jump speed optimizen

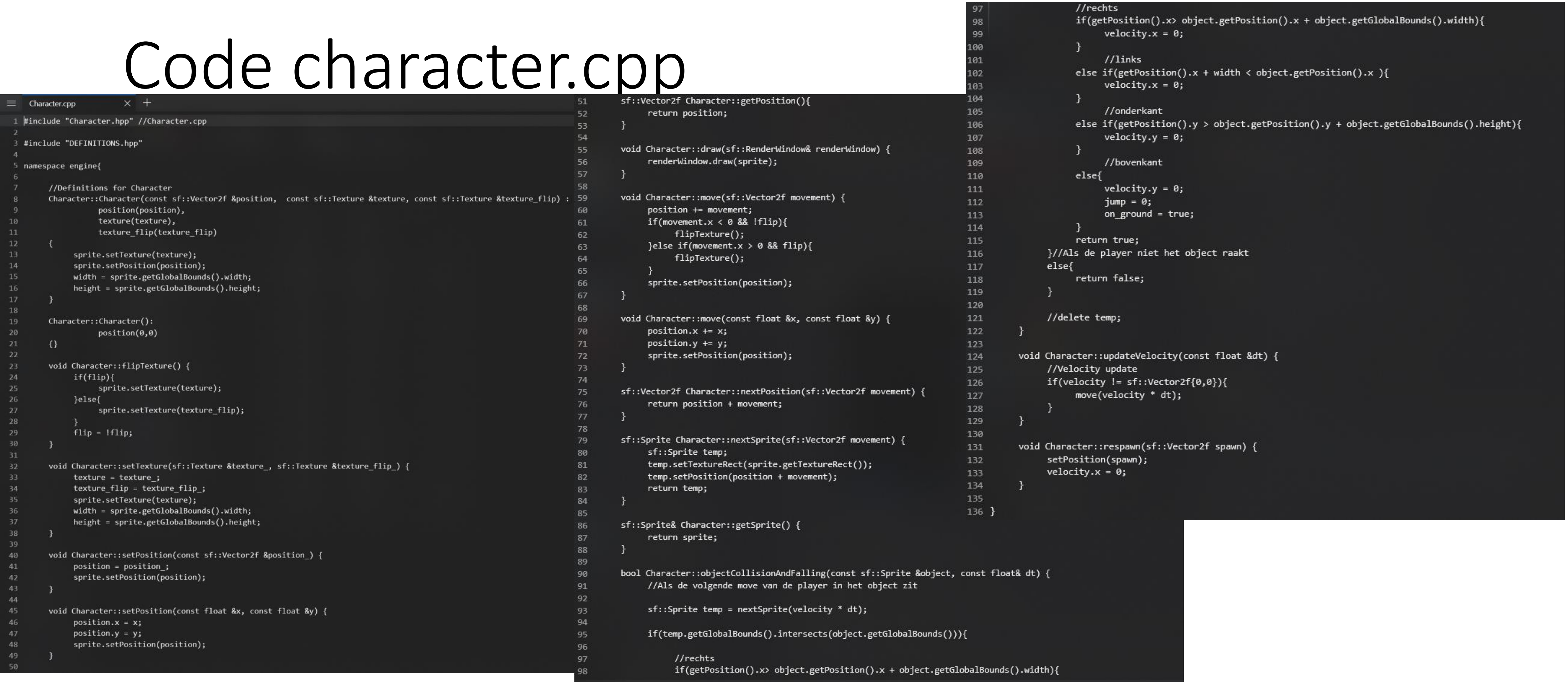
in lijst Done

LEDEN: D JK R +

Omschrijving Bewerken

Jump speed en Gravity goed instellen, zodat het natuurlijk aanvoelt

Code character.cpp



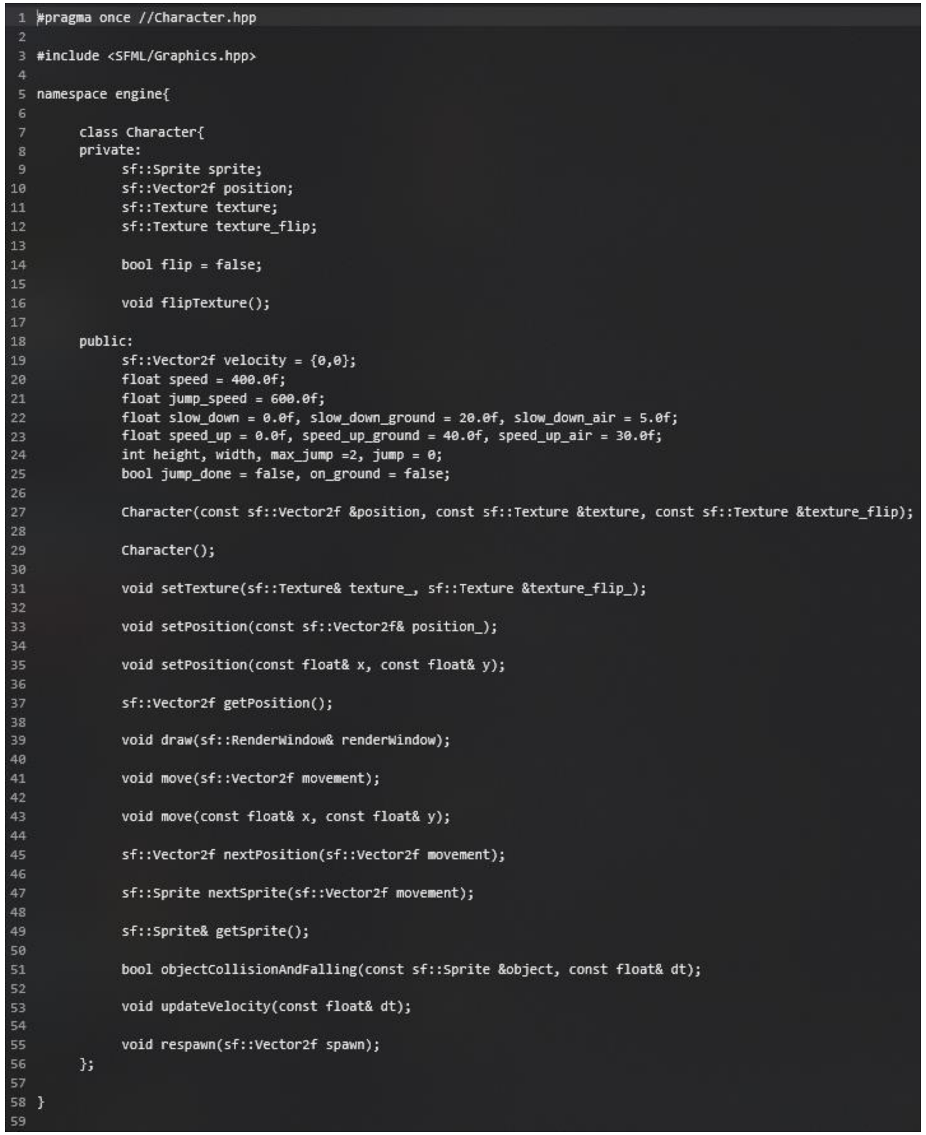
```cpp
#include "Character.hpp" //Character.cpp

#include "DEFINITIONS.hpp"

namespace engine{

    //Definitions for Character
    Character::Character(const sf::Vector2f &position,  const sf::Texture &texture, const sf::Texture &texture_flip) :
            position(position),
            texture(texture),
            texture_flip(texture_flip)
    {
        sprite.setTexture(texture);
        sprite.setPosition(position);
        width = sprite.getGlobalBounds().width;
        height = sprite.getGlobalBounds().height;
    }

    Character::Character():
            position(0,0)
    {}

    void Character::flipTexture() {
        if(flip){
            sprite.setTexture(texture);
        }else{
            sprite.setTexture(texture_flip);
        }
        flip = !flip;
    }

    void Character::setTexture(sf::Texture &texture_, sf::Texture &texture_flip_) {
        texture = texture_;
        texture_flip = texture_flip_;
        sprite.setTexture(texture);
        width = sprite.getGlobalBounds().width;
        height = sprite.getGlobalBounds().height;
    }

    void Character::setPosition(const sf::Vector2f &position_) {
        position = position_;
        sprite.setPosition(position);
    }

    void Character::setPosition(const float &x, const float &y) {
        position.x = x;
        position.y = y;
        sprite.setPosition(position);
    }

```

```cpp
    sf::Vector2f Character::getPosition(){
        return position;
    }

    void Character::draw(sf::RenderWindow& renderWindow) {
        renderWindow.draw(sprite);
    }

    void Character::move(sf::Vector2f movement) {
        position += movement;
        if(movement.x < 0 && !flip){
            flipTexture();
        }else if(movement.x > 0 && flip){
            flipTexture();
        }
        sprite.setPosition(position);
    }

    void Character::move(const float &x, const float &y) {
        position.x += x;
        position.y += y;
        sprite.setPosition(position);
    }

    sf::Vector2f Character::nextPosition(sf::Vector2f movement) {
        return position + movement;
    }

    sf::Sprite Character::nextSprite(sf::Vector2f movement) {
        sf::Sprite temp;
        temp.setTextureRect(sprite.getTextureRect());
        temp.setPosition(position + movement);
        return temp;
    }

    sf::Sprite& Character::getSprite() {
        return sprite;
    }

    bool Character::objectCollisionAndFalling(const sf::Sprite &object, const float& dt) {
        //Als de volgende move van de player in het object zit

        sf::Sprite temp = nextSprite(velocity * dt);

        if(temp.getGlobalBounds().intersects(object.getGlobalBounds())){

            //rechts
            if(getPosition().x> object.getPosition().x + object.getGlobalBounds().width){
```

```cpp
            //rechts
            if(getPosition().x> object.getPosition().x + object.getGlobalBounds().width){
```

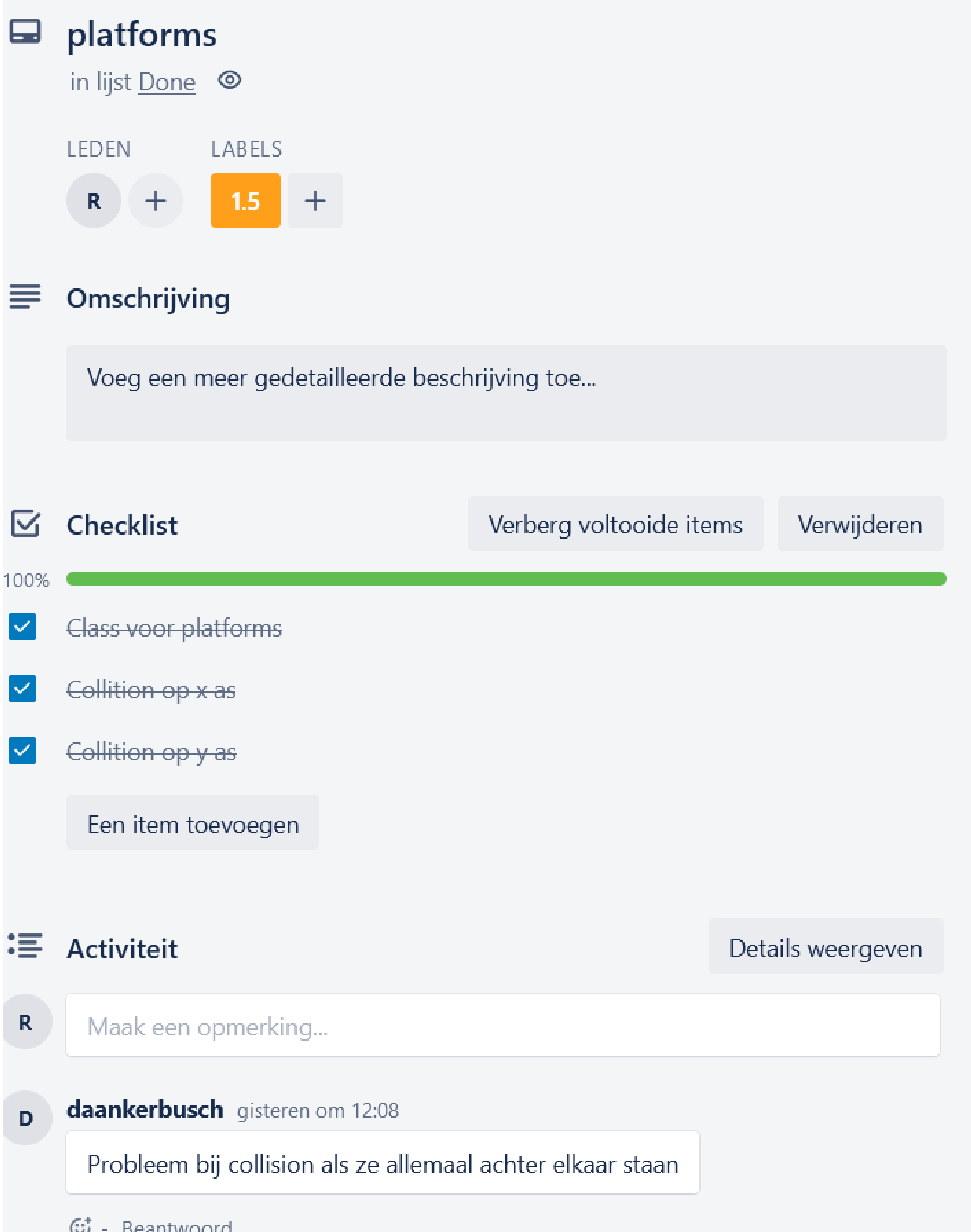
```cpp
                velocity.x = 0;
            }
                //links
            else if(getPosition().x + width < object.getPosition().x ){
                velocity.x = 0;
            }
                //onderkant
            else if(getPosition().y > object.getPosition().y + object.getGlobalBounds().height){
                velocity.y = 0;
            }
                //bovenkant
            else{
                velocity.y = 0;
                jump = 0;
                on_ground = true;
            }
            return true;
        }//Als de player niet het object raakt
        else{
            return false;
        }

        //delete temp;
    }

    void Character::updateVelocity(const float &dt) {
        //Velocity update
        if(velocity != sf::Vector2f{0,0}){
            move(velocity * dt);
        }
    }

    void Character::respawn(sf::Vector2f spawn) {
        setPosition(spawn);
        velocity.x = 0;
    }

}
```

Code character.hpp



```cpp
#pragma once //Character.hpp

#include <SFML/Graphics.hpp>

namespace engine{

    class Character{
    private:
        sf::Sprite sprite;
        sf::Vector2f position;
        sf::Texture texture;
        sf::Texture texture_flip;

        bool flip = false;

        void flipTexture();

    public:
        sf::Vector2f velocity = {0,0};
        float speed = 400.0f;
        float jump_speed = 600.0f;
        float slow_down = 0.0f, slow_down_ground = 20.0f, slow_down_air = 5.0f;
        float speed_up = 0.0f, speed_up_ground = 40.0f, speed_up_air = 30.0f;
        int height, width, max_jump =2, jump = 0;
        bool jump_done = false, on_ground = false;

        Character(const sf::Vector2f &position, const sf::Texture &texture, const sf::Texture &texture_flip);

        Character();

        void setTexture(sf::Texture& texture_, sf::Texture &texture_flip_);

        void setPosition(const sf::Vector2f& position_);

        void setPosition(const float& x, const float& y);

        sf::Vector2f getPosition();

        void draw(sf::RenderWindow& renderWindow);

        void move(sf::Vector2f movement);

        void move(const float& x, const float& y);

        sf::Vector2f nextPosition(sf::Vector2f movement);

        sf::Sprite nextSprite(sf::Vector2f movement);

        sf::Sprite& getSprite();

        bool objectCollisionAndFalling(const sf::Sprite &object, const float& dt);

        void updateVelocity(const float& dt);

        void respawn(sf::Vector2f spawn);
    };

}
```

Platforms gemaakt



platforms

in lijst Done

LEDEN

R +

LABELS

1.5 +

Omschrijving

Voeg een meer gedetailleerde beschrijving toe...

Checklist

100%

☑ ~~Class voor platforms~~

☑ ~~Collition op x as~~

☑ ~~Collition op y as~~

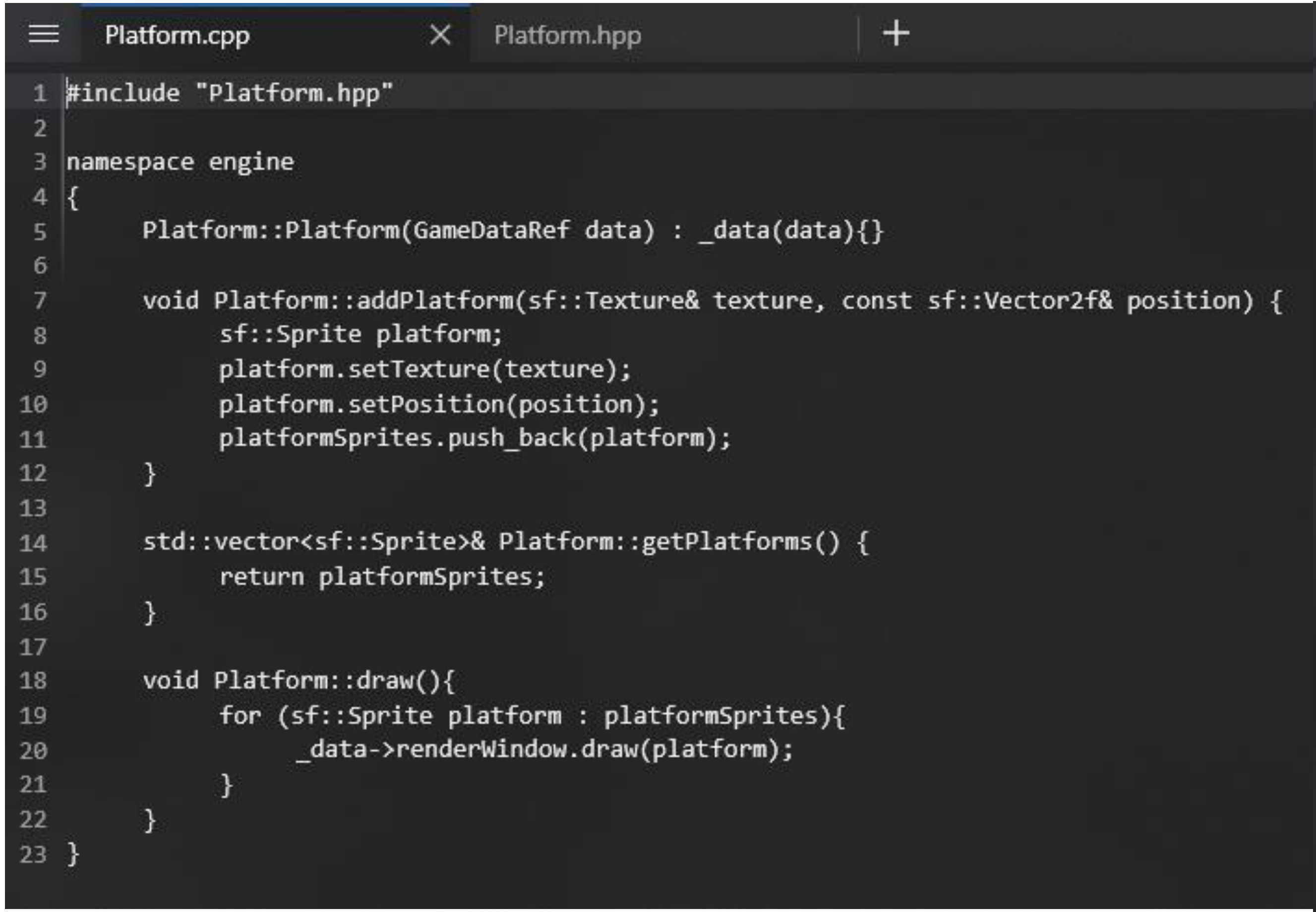
Activiteit

R Maak een opmerking...

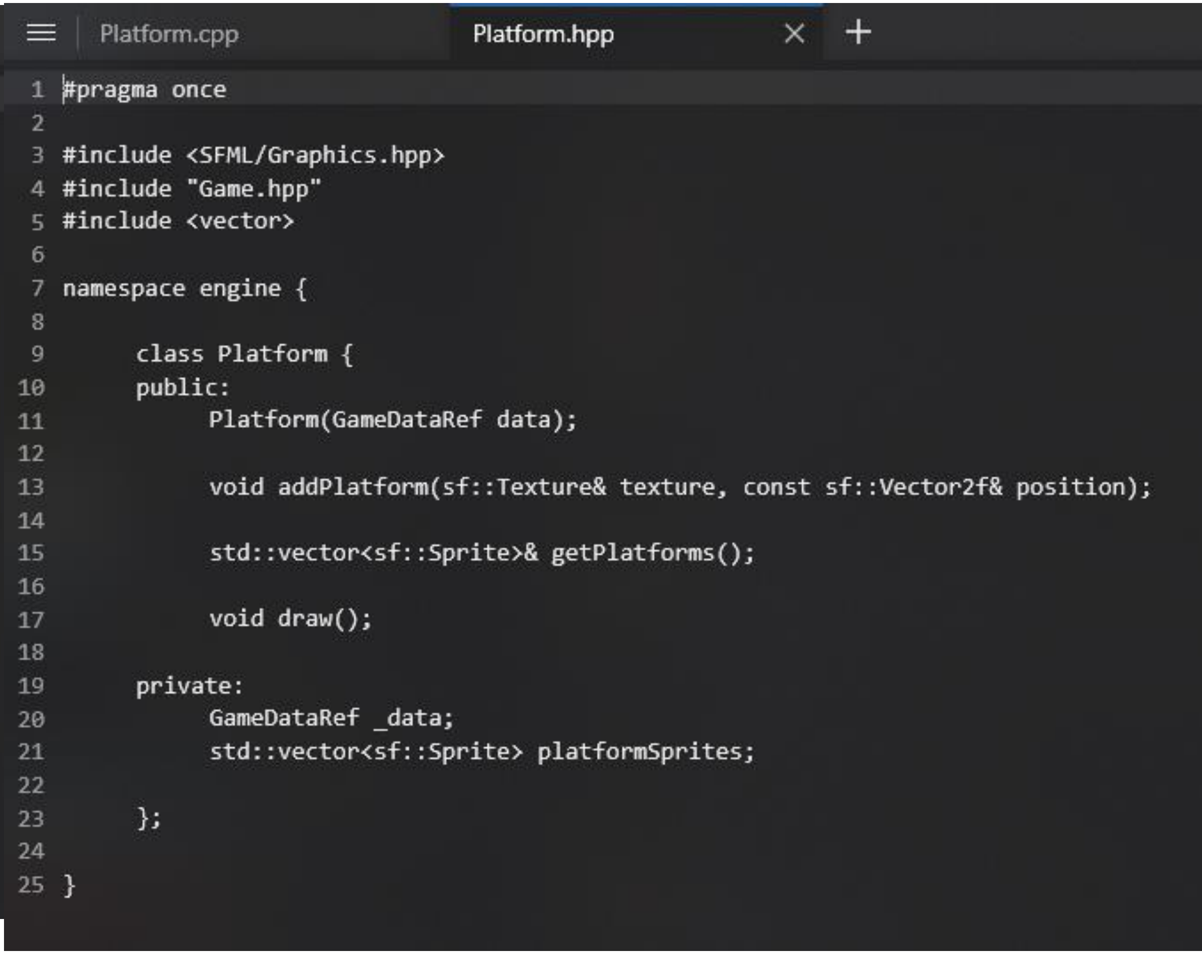
D **daankerbusch** gisteren om 12:08

Probleem bij collision als ze allemaal achter elkaar staan

Code



```cpp
#include "Platform.hpp"

namespace engine
{
    Platform::Platform(GameDataRef data) : _data(data){}

    void Platform::addPlatform(sf::Texture& texture, const sf::Vector2f& position) {
        sf::Sprite platform;
        platform.setTexture(texture);
        platform.setPosition(position);
        platformSprites.push_back(platform);
    }

    std::vector<sf::Sprite>& Platform::getPlatforms() {
        return platformSprites;
    }

    void Platform::draw(){
        for (sf::Sprite platform : platformSprites){
            _data->renderWindow.draw(platform);
        }
    }
}
```



```cpp
#pragma once

#include <SFML/Graphics.hpp>
#include "Game.hpp"
#include <vector>

namespace engine {

    class Platform {
    public:
        Platform(GameDataRef data);

        void addPlatform(sf::Texture& texture, const sf::Vector2f& position);

        std::vector<sf::Sprite>& getPlatforms();

        void draw();

    private:
        GameDataRef _data;
        std::vector<sf::Sprite> platformSprites;

    };

}
```

Extra taken

- Ik heb de achtergrond gemaakt
- En daan + joris geholpen bij het maken van testlevel