8.2 C# SFML - Animation in SFML

SFML biete selber keine Objekte, um Animation zu vereinfachen. Deswegen arbeiten wir mit einer Liste. Als Spritesheet nutzen wir die "spriteSheet46x50.png".

Als erstes müssen wir wie im vorherigen Kapitel die Textur erst laden.

Wir erstellen ein Texture Objekt in Mainscreen:

public Texture spritesheet;

// Setup, wird immer einmal zu Beginn eines Screens aufgerufen

// Hier Startwerte setzen!

public override void setup()

{

spritesheet = new Texture("assets/spriteSheet46x50.png");

Dafür muss die "spriteSheet46x50.png" natürlich im assets Ordner liegen und die "kopieren wenn neuer" Eigenschaft eingestellt sein.

Dann können wir in unserer Entity, schon mal eine Sprite Liste erstellen, in die alle Sprites unserer Animation geladen werden.

public List<Sprite> animation;

public Sprite graphic;

MainScreen screen;

// Konstruktor

public ExampleEntity(MainScreen parentScreen)

{

screen = parentScreen;

animation = new List<Sprite>();

Jetzt müssen wir die Liste füllen, testweise erstellen wir erstmal ein Sprite und packen es in die Liste. Außerdem sagen wir, dass unsere Graphic das erste Sprite der Liste sein soll.

Sprite nSprite = new Sprite(screen.spritesheet);

animation.Add(nSprite);

graphic = animation[0];



Jetzt haben wir das gesame SpriteSheet als Sprite, wir wollen aber nur einen Teil davon. Deswegen müssen wir beim Erstellen des Sprites neben der Textur, noch ein **IntRect** angeben.

Statt:  
Sprite nSprite = new Sprite(screen.spritesheet);

jetzt:

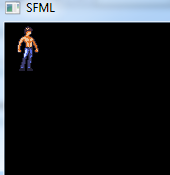
Sprite nSprite = new Sprite(screen.spritesheet, new IntRect(0,0,0,0));

Die vier Zahlen des IntRects stehen für X-Position,Y-Position,Breite,Höhe in dieser Reihenfolge. Die Breite und Höhe des Teilbildes ist im Dateinamen des SpriteSheets angegeben, 46x50.

(Wenn das nicht der Fall ist, muss man nachsehen/nachzählen)

Sprite nSprite = new Sprite(screen.spritesheet, new IntRect(0,0,46,50));

Da als X/Y 0 angegeben ist, nehmen wir den Teil der Spritesheet ganz oben links. Das Ergebnis:



Sieht schonmal gut aus. Jetzt wollen wir mehrere Sprites in die Liste laden, und zwar immer andere. Dazu ändern wir immer die X/Y Position. Am einfachsten geht dies durch eine For Schleife:

for (int i = 0; i < 8; i++)

{

Sprite nSprite = new Sprite(screen.spritesheet, new IntRect(46\*i,0,46,50));

animation.Add(nSprite);

}

Das "46\*i" sorgt dafür, dass wir jeden Durchlauf ein Bild 46 Pixel weiter rechts nehmen(die breite eines bildes).

Jetzt müssen wir nurnoch die Graphic nacheinander die verschiedenen Sprites anzeigen lassen.

Dafür brauchen wir einen typischen Counter, eine Int-Variable.

public List<Sprite> animation;

public Sprite graphic;

int currentFrame = 0;

MainScreen screen;

Diesen lassen wir in der Loop hochzählen, aber sobald er die 8 erreicht, soll er wieder auf 0 gesetzt werden.(Die Anzahl sollte immer gleich sein zu der For schleife im Konstruktor!)

// Loop Methode

public void loop()

{

currentFrame++;

if (currentFrame >= 8)

{

currentFrame = 0;

}

}

Jetzt lassen wir die graphic in der Loop immer auf die aktuelle Animationssprite zeigen:

// Loop Methode

public void loop()

{

currentFrame++;

if (currentFrame >= 8)

{

currentFrame = 0;

}

graphic = animation[currentFrame];

}

Da wir mit 60 FPS arbeiten, ist die Animationsgeschwindigkeit sehr hoch. Um diese besser zu kontrollieren, brauchen wir noch einen Counter.

public List<Sprite> animation;

public Sprite graphic;

int currentFrame = 0;

int frameCounter = 0;

MainScreen screen;

Diesen lassen wir wieder genauso hochzählen in der loop, allerdings diesmal bis 5.

// Loop Methode

public void loop()

{

currentFrame++;

if (currentFrame >= 8)

{

currentFrame = 0;

}

frameCounter++;

if (frameCounter >= 5)

{

frameCounter = 0;

}

graphic = animation[currentFrame];

}

Wenn wir jetzt "currentFrame++" in die if-Klammer des Frame Counters tun, können wir die Animation um den Faktor 5 verlangsamen.

// Loop Methode

public void loop()

{

frameCounter++;

if (frameCounter >= 5)

{

currentFrame++; // <--- hier einfügen

frameCounter = 0;

}

//currentFrame++; <-- hier entfernen

if (currentFrame >= 8)

{

currentFrame = 0;

}

graphic = animation[currentFrame];

}

Jetzt haben wir eine schöne Animation.

Achtung! Wenn man Position/Rotation/Scale etc. ändern möchte:

Sollte man diese Werte wie hier übernehmen, bevor man das Sprite wechselt:

// Loop Methode

public void loop()

{

frameCounter++;

if (frameCounter >= 5)

{

currentFrame++;

frameCounter = 0;

}

if (currentFrame >= 8)

{

currentFrame = 0;

}

animation[currentFrame].Position = graphic.Position;

animation[currentFrame].Rotation = graphic.Rotation;

animation[currentFrame].Scale = graphic.Scale;

graphic = animation[currentFrame];

}

Da sonst nur einzelne Frames geändert werden, aber nicht die ganze Animation!