

Génération d’animations

Compilateur - Cahier des charges

6 décembre 2017

Auteur : Lovis Thomas,

Professeur : Tièche François

Table des matières

[1 Introduction 2](#_Toc500144159)

[1.1 Configuration 2](#_Toc500144160)

[1.1.1 Machine physique 2](#_Toc500144161)

[1.1.2 Machine virtuelle 2](#_Toc500144162)

[2 Prérequis 3](#_Toc500144163)

[2.1 Téléchargement 3](#_Toc500144164)

[2.2 Décompression 3](#_Toc500144165)

[3 Paramétrage 4](#_Toc500144166)

[3.1 .Config 4](#_Toc500144167)

[3.2 Options de compilations 4](#_Toc500144168)

[3.2.1 Prérequis 4](#_Toc500144169)

[3.2.2 Manuel 4](#_Toc500144170)

[3.2.3 Menu de choix des modules 5](#_Toc500144171)

[3.2.4 Choix des modules 5](#_Toc500144172)

[4 Exécution 7](#_Toc500144173)

[4.1 Compilation 7](#_Toc500144174)

[4.2 Installation 7](#_Toc500144175)

[5 Conclusion 8](#_Toc500144176)

[5.1 Résultats 8](#_Toc500144177)

[5.1.1 Version du noyau 8](#_Toc500144178)

[5.1.2 Vérification des modules 8](#_Toc500144179)

[5.2 Sources 9](#_Toc500144180)

# Introduction

Durant la 3ème année du semestre d’automne en développement logiciel et multimédia, les élèves sont chargés de reprendre les travaux faits en cours et inventer un nouveau langage (compilable ou interprétable selon le sujet).

## But fixé

Le but de notre projet et de créer un langage qui va générer des animations. L’idée est de simplifier quelques formes en 2D dimensions, pouvoir leur donner une texture et éventuellement une notion de Sprite. Dans tous les cas, les objets ainsi crées devront pouvoir s’animer avec des mouvements (translation, rotation). Dans une version avancée on pourrait même y ajouter de la physique.

## Programme et matériel

Le langage des différents programmes (les analyseurs, le couseur et l’interpréteur) seront écrits en python.

La librairie utilisée pour la génération de l’environnement graphique sera le module « Pygame ».

## Exemple pré-développement d’utilisation

Voyons un exemple du code qu’on l’on aimerait écrire pour fournir une animation :

ball1 = Ball(0, 0, 50); //posx, posy, radius

ball2 = Ball(100, 0, 30);

moveUp = Move(0, 1); //On bouge de +1 en y

rotationClock = Rotate(π/8);

timer = Timer(10’000); //Temps en ms

Start(timer)

While(timer > 0){

Animate(ball1, moveUp);

Animate(ball2, rotationClock);

}

//A voir Timer pas forcément optimal on pourrait définir un EndTime

# Développement

## Fonctionnalités implémentées

### Les formes

### Les animations

## Analyseur lexicale

TODO

## Analyser syntaxique

TODO

## Interprétation

TODO

# Prise en main

TODO Fichier d’exemple et explication

# Conclusion

## Sources