Departamento de Eletrónica, Telecomunicações e Informática

Compiladores

Scratch

Autores:

André Brandão 84916
Rafael Teixeira 84746
Daniel Nunes 84793
Rafael Direito 84921
Pedro Ferreira 84735

Docentes:

Miguel Oliveira e Silva
André Zúquete

28 de Junho de 2018

Universidade de Aveiro

Conteúdo

Lista de Figuras

Introdução

Inserido no plano curricular da disciplina de Compiladores, do curso de Engenharia Informática, da Universidade de Aveiro e leccionada pelos professores Miguel Silva e André Zúquete, este relatório tem como objetivo explicar os conceitos utilizados para elaborar o projeto e como é que foram implementados. Antes de introduzir a descrição do trabalho, refere-se que o tema escolhido não consta na lista fornecida pelo professor. O tema baseia-se numa versão diferente do scratch, onde agora não existe interface gráfica. As Personagens/Sprites são definidas (tamanho, cor, caminho para a imagem/textura, etc.) num ficheiro de configuração à parte com a informação estruturada e o código que executa a animação é feito em C++. Para o backend do compilador utilizámos C++ com a livraria SFML (https://www.sfml-dev.org/) que permite o desenvolvimento de aplicações multimédia, jogos, etc... multi-plataforma.

Funcionalidades

- Definição de variáveis com scope.
- Definição de funções.
- Instruções condicionais (if, elsif e else)
- Criação de classes.
- Saber se o utilizador está a pressionar alguma tecla.
- Criação de objetos de classes pre-definidas e executar métodos sobre eles.

Documentação

Atribuições de valor

As atribuições de valor são feitas de um modo bastante comum:

```
DataType varName = expression;
```

Exemplos:

```
num a = 1;
num b = 1+2*(1+3)/5;
string s = "olá";
Sprite s = create Sprite("spriteConfig.sprite");
```

Loops

```
While loop:
    while(expression) {
        statList
    }
```

Exemplos:

```
While loop:
    while (true) {
        num a = 1;
    }
    while (a < 2 || 1 > 4 && a == 0) {}
For loop:
    for (expression;expression;expression) {
        statList
    }
    exempos:
        for (num i = 0; i < 10; n = n + 1) {}
```

Instruções Condicionais

```
if (expression) {
    statList
} elsif (expression) {
    statList
} else {
    statList
}
```

Funções

A definição das funções é feita da seguinte maneira:

```
function DataType ID(argsList) {
    statList
}

exemplo:
    function List function1(Sprite s, num a, string b) {
        List l;
        give l;
    }
```

Operações

- Potência. (exemplo: $2\hat{3}$).
- Multiplicação. (exemplo: 2*3).
- Divisão. (exemplo: 2/3).
- Soma. (exemplo: 2+3).
- Subtração. (exemplo: 2-3).
- Comparadores. (exemplo: a == b, a != b, a ¡ b, a ¿ b, a ¡= b, a ¿= b)
- Comparadores lógicos. (exemplo: a and b, c or d)
- Expressão com sinais. (exemplo: -1, -(1+2))

Classes e métodos

Window - janela ambiente do jogo

void close()	fecha a janela.
num getHeight()	retorna a altura da janela.
num getWidth()	retorna a largura da janela.
num getPosition()	retorna a posição da janela.
void setSize(num x, num y)	atribuí o tamanho da janela.
void setTitle(String s)	atribuí um título à janela.

Sprite - objeto com um atributo, por exemplo, um quadrado com uma textura

void move(num x, num y)	mexe o Sprite para uma nova posição de uma forma relativa.
void goTo()	mexe o Sprite para uma nova posição de uma forma absoluta.
void rotate(num degrees)	roda o Sprite em degrees.
void setSize(num h, num w)	atribuí um novo tamanho à Sprite.
boolean touching(Sprite s)	verifica se o Sprite está a colidir com outro objeto.
void setGravity(num g)	atribuí uma nova gravidade.
void changeCostume(String nome)	altera o antigo background.
num getX()	retorna o valor X da posição do Sprite.
num getY()	retorna o valor Y da posição do Sprite.

List - estrutura de dados que guarda Sprites

void insert(Sprite s)	insere a Sprite na última posição da estrutura de dados.
void insert(num pos, Sprite s)	insere uma Sprite na posição pos da estrutura de dados.
void remove(Sprite s)	remove a Sprite na última posição da estrutura de dados.
void remove(num pos)	remove a Sprite na posição pos da estrutura de dados.
Sprite get(num pos)	retorna a Sprite na posição pos da estrutura de dados.

Biblioteca de apoio C++

As classes usadas na linguagem c++ foram baseadas em classes existentes de SFML, desenvolvemos 4 classes contudo só conseguimos implementar duas delas para serem usadas no código final (ScratchTexture e ScratchSprite), as outras duas sofriam de herança múltipla o que não era contemplado pela nossa linguagem sendo que nos impossibilitou de realizar a execução de métodos nas mesmas (era necessário acrescentar a deteção de métodos ambíguos e a adição de prefixo de desambiguação). Retirando estas duas classes com as outras duas seria possível realizar animações simples de movimento de Sprite.

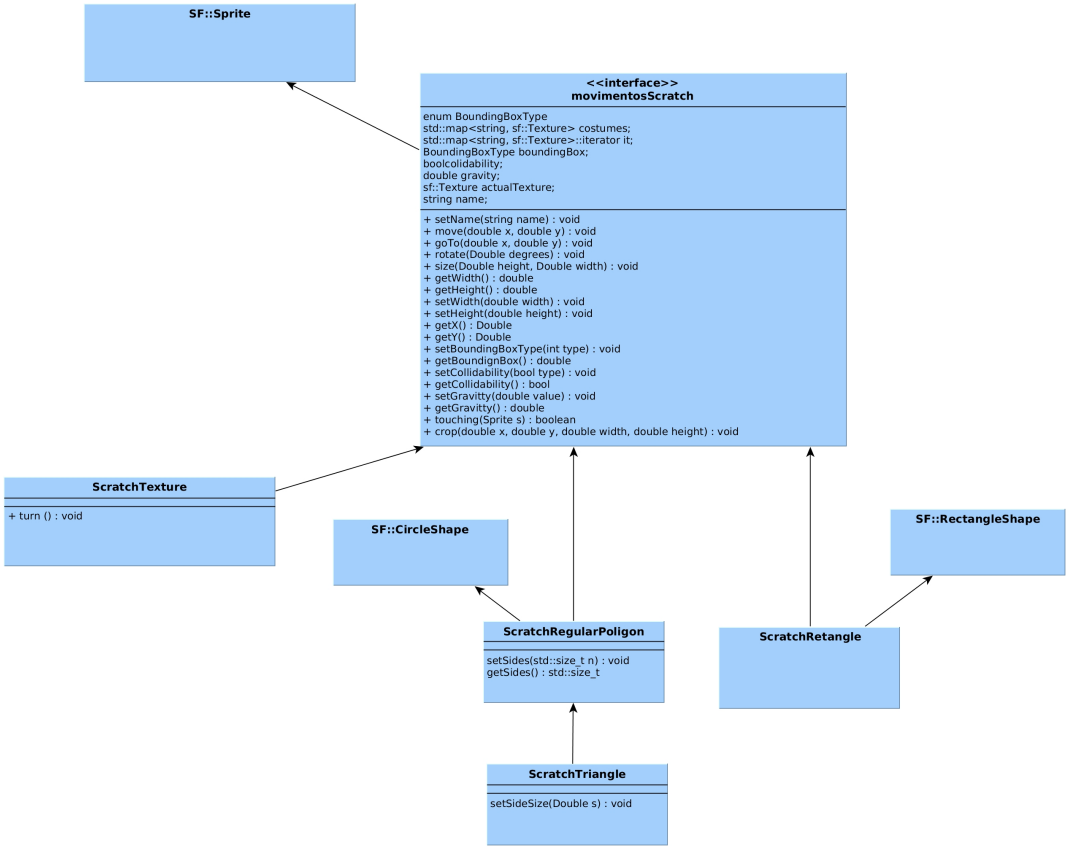

Figura 1: Diagrama UML.

Classificação

Pontuação	Tarefas	Autor
40	Livraria de apoio C++	Rafael Teixeira.
40	Interpretador do Sprite	Rafael Teixeira.
40	Gramática Sprite	Rafael Teixeira (20), Pedro Ferreira (20).
80	SpriteSemanticCheck	Pedro Ferreira.
50	Gramática Scratch	André Brandão (25), Rafael Direito (25).
100	Compilador Scratch	Rafael Direito (75), Daniel Nunes (25).
150	ScratchSemanticCheck	André Brandão (75), Daniel Nunes (75).

Conclusão

Olhando em retrospectiva, consideramos que os objetivos principais relativos a este trabalho foram atingidos, sendo que outros ficaram por implementar/testar. Através dos vários exemplos em anexo, é possível observar as diversas funcionalidades implementadas. Em suma, concluímos que, com este trabalho, os conhecimentos sobre ANTLR4, String Template, C++, Java e SFML foram, sem qualquer dúvida, melhorados e que a capacidade de pesquisa e autonomia foram, também elas, bem desenvolvidas, na tentativa de obter soluções face aos problemas encontrados.

Referências

[1] Slides das aulas teóricas fornecidos no elearning.