Programação Orientada a Objetos - 1ª meta

Kylix Afonso a2020146228@isec.pt

Gonçalo Salgueirinho a2020142627@isec.pt

17 de janeiro de 2022

1 Introdução

Este documento irá servir de justificação e explicação para as estratégias, mecanismos e classes escolhidas e implementadas no nosso trabalho prático de Programação Orientada a Objetos e incluíra também um breve guia de uso do programa até agora concretizado. Sem querer baixar as expectativas do professor responsável pela nossa defesa, queremos mencionar que o programa está num momento incrivelmente inicial e infelizmente, devido à falta de tempo da nossa parte, não conseguimos implementar imensas coisas que já temos planeadas há algum tempo.

2 Relatório

2.1 Funcionalidades implementadas

- Definição das classes necessárias ao funcionamento do simulador
  - Interface
  - Jogo
  - Recursos
  - Ilha
  - Zona
  - Edifício
  - Trabalhador
- Arquitetura simples e organizada
- Interface completa
  - Menu de ajuda
  - Ilha apresentada da forma pedida
  - Apresentação de outras informações uteis (dia, momento do dia, recursos, etc)
- Comandos implementados
  - Relacionados com o jogo (next e exit)
  - Relacionados com ficheiros (exec e config)
  - Relacionados com edifícios (cons, des, levelup e vende)
  - Relacionados com trabalhadores (cont, move)
  - Relacionados com recursos (vende)
  - Relacionados com zonas (list)
  - Debug (debcash, debed e debkill)

2.2 Justificação para as principais escolhas de implementação

- Organização do source code
  - O código do trabalhado está distribuído por vários diretórios de forma a facilitar a sua leitura. Os ficheiros diretamente relaciondos com classes encontram-se em diretórios com a primeira letra do nome em maiúscula.
- Interface
  - Como um dos nossos objetivos inicialmente era implementar uma interface gráfica, o input e output são maioritariamente através de um ambiente feito no terminal, onde a interface acede a uma parte do jogo que é destinada para isso. Desta forma, mais tarde poderá ser implementada uma interface em qt de forma mais fácil.

2.3 Classes principais

- Jogo: Classe superior. Trata do todo o funcionamento do jogo.
  - Interpreta os comandos
  - Simula as manhãs e tardes
- Zona: é a classe mais importante do jogo, contém informação como trabalhadores, edificio, recursos, entre outros. Tipos de Zona:
  - Deserto;
  - Floresta;
  - Pântano;
  - Pastagem;
  - Montanha;
  - ZonaX.
- Ilha: contém um array bidimensional de Zonas, é a classe que estabelece a principal relação visual entre o utilizador e o jogo.
- Edifício: especializado para cada função (produção de recursos).
  - Mina de ferro;
  - Mina de carvão;
  - Central elétrica;
  - Bateria;
  - Fundição;
  - EdifícioX.
- Trabalhador: tipos de trabalhador (subclasses onde iremos aplicar polimorfismo):
  - Lenhador;
  - Mineiro;
  - Operador;
- Interface: classe que trata da maior parte do relação input / output entre o utilizador e o jogo.
- Recursos: classe simples que contém apenas a quantidade de um certo tipo de recurso
  - Dinheiro
  - Madeira
  - Ferro
  - Aço
  - Vigas de madeira
  - Carvão
  - Eletricidade

2.4 Problemas

- Problemas existentes no trabalho:
  - Pequenos detalhes como o descanço dos trabalhadores.
  - Não implementação de saves e loads. Não é possivel guardar em memória nem carregar da mesma um jogo que ainda a decorrer.
  - ZonaX sem qualquer tipo de caracteristica. Não acrescenta nada à funcionalidade do simulador.
  - EdificioX também sem qualquer tipo de funcionalidade.
- Por falta de organização e responsabilidade não conseguimos terminar o trabalho no que toca aos aspetos a cima.

3 Guia de uso prático do GAMEpoo

3.1 Makefile

O Makefile usado neste projeto é muito simples e só existe para que seja possível compilar o projeto em ambiente Linux (existe um problema de portabilidade do programa que implica a impossibilidade de utilização do programa nos dois sistemas operativos sem a alteração de todas as chamadas da função de rand() para random() e vice-versa, isto causa um grande problema que é o facto de ambos usarmos ambiente Linux e o projeto necessariamente ter que ser portado para CLion que também existe no Linux mas está originalmente desenhado para ambiente Windows, sendo assim, na defesa, caso seja possível, iremos usar este argumento para possibilitar a mudança desses dois caráteres, o "o"e o "m"de random para que possamos fazer a defesa no nosso ambiente de eleição e o professor poder ver o programa a correr em Windows à mesma). A utilização deste Makefile implica só usar o comando "make", o Makefile aqui usado faz sempre relink de todos os source files e object files (isto é propositado).

3.2 Programa

O programa tem uma interface simples. Iniciando o executável, é de notar que aparecem logo 2 opções, jogar e instruções, jogar levará ao inicio da simulação e instruções mostrará o conteúdo de um ficheiro denominado de instructions. Selecionando "jogar", após escolher o tamanho da ilha, será possível simular o mundo definido no trabalho prático, sendo que aparecerá de imediato um mapa com todas as zonas, os trabalhadores naquelas zonas e os edíficios neles colocados. A partir daí o jogo funciona através de comandos/ordens. O simulador termina quando acabarem os recursos e os trabalhadores.

Referências

As nossas principais fontes de referência são as aulas práticas e teóricas da cadeira, no nosso caso lecionadas pelos professores Ivo Gonçalves e João Durães, respetivamente.