UNIVERSIDADE FEDERAL DE VIÇOSA - UFV - CAMPUS FLORESTAL

ENGENHARIA DE SOFTWARE II

Especificação de Classes e Funcionalidades da Sprint 03 para os Desenvolvedores Juniores

Projeto Integrador - Equipe 2º ano

Florestal - MG

2024

SUMÁRIO

1. Objetivo

Este documento tem como finalidade fornecer uma visão detalhada e clara das classes que compõem o sistema, conforme o planejamento da Sprint 03. Ele apresenta as responsabilidades, atributos e métodos de cada classe, facilitando a implementação e integração do código.

O objetivo principal é garantir que todos os desenvolvedores juniores tenham uma compreensão unificada da estrutura do sistema, assegurando que os padrões de orientação a objetos sejam seguidos e promovendo a qualidade do software.

2. Classes de Entidade

2.1 Classe Nível

- **Propósito**

  A classe abstrata Nível define os elementos básicos de um nível no sistema, como pontuação, tempo e penalidades. É a base para outras classes específicas de níveis.

- **Atributos**

  - **pontuacaoInicial : int** — Pontuação inicial do nível.
  - **tempoMinimo : float** — Tempo mínimo necessário para concluir o nível.
  - **tempoMaximo : float** — Tempo máximo permitido para concluir o nível.
  - **bonusTempo : int** — Uma pontuação a mais para quando o nível é terminado em menos tempo que o mínimo.
  - **penalidadeTempo : int** — É uma pontuação a menos para quando o nível é terminado em mais tempo que o máximo.
  - **erroMaximo : int** — O máximo de erros que vamos tirar ponto. Mais erros que isso não tira ponto, para o Aluno não perder pontos demais se estiver com muita dificuldade.
  - **penalidadeErro : int** — Penalidade em pontuação por erro.

- **Métodos**

  - **Construtor:** : Inicializa todos os atributos da classe Nível, incluindo a pontuação inicial, os tempos mínimo e máximo, o bônus e penalidades de tempo, o número máximo de erros e as penalidades por erro, garantindo que o objeto esteja configurado corretamente com os valores iniciais para o funcionamento do nível.

  - **Métodos específicos:**
    - **calcularPontuacao(int qtdErros, float tempoGasto)** — Este método calcula a pontuação final de um aluno com base no

número de erros cometidos e no tempo gasto. A pontuação é ajustada por meio de uma soma que inclui bônus ou penalidades, já que as penalidades possuem valores negativos. As regras são as seguintes:

1. **Penalidade por erros:**
   a. Para cada erro cometido, aplica-se uma penalidade calculada como penalidadeErro * qtdErros. Caso qtdErros exceda o valor de erroMaximo, **a penalidade será limitada** a penalidadeErro * erroMaximo.

2. **Bônus ou Penalidade por Tempo:**
   a. Se o tempo gasto (**tempoGasto**) for menor ou igual a **tempoMinimo**, soma-se um bônus equivalente a **bonusTempo**.
   b. Se o tempo gasto for maior que **tempoMaximo**, soma-se uma penalidade equivalente a **penalidadeTempo**.

3. **Dica de Implementação:**
   a. Como as penalidades já possuem valores negativos, a soma resultante ajustará a pontuação de forma correta sem necessidade de subtrações adicionais. Por exemplo, a fórmula pode ser implementada diretamente como **pontuacaoFinal = pontuacaoInicial + penalidadeErro + bonusTempo**.
4. **Retorno**
   a. O método retorna a pontuação final ajustada, sem alterar o valor original de **pontuacaoInicial**.

- **Métodos Get e Set:** Métodos para acessar e modificar os valores dos atributos:
  - **getPontuacaoInicial() : int** — Retorna a pontuação inicial do nível.
  - **getTempoMinimo() : float** — Retorna o tempo mínimo necessário.
  - **getTempoMaximo() : float** — Retorna o tempo máximo permitido.
  - **getBonusTempo() : int** — Retorna o bônus em pontuação por tempo economizado.
  - **setBonusTempo(int bonusTempo)** — Define o valor do bônus em pontuação a ser considerado quando o nível é completado em menos tempo que o mínimo. Este método apenas configura o valor do bônus, não altera a

pontuação diretamente.

- **getPenalidadeTempo() : int** — Retorna a penalidade em tempo por erro.

- **setPenalidadeTempo(int penalidadeTempo)** — Define a penalidade em tempo.

- **getErroMaximo() : int** —Retorna o número máximo de erros permitidos para descontar pontos.

- **getPenalidadeErro() : int** — Retorna a penalidade em pontuação por erro.

- **setPenalidadeErro(int penalidadeErro)** — Define a penalidade por erro.

2.2 Classe Fase

- **Propósito**

  Define um conjunto de níveis que compõem uma fase do jogo, fornecendo uma estrutura para progresso e organização.

- **Atributos**

  - **niveis : ArrayList<Nivel>** — Lista de níveis da fase.
  - **explicacao : String** — Texto explicativo sobre a fase.

- **Métodos**

  - **Construtor:** Inicializa a lista de níveis da fase e define a explicação associada à fase, configurando os atributos necessários para a organização e o progresso do jogo dentro dessa fase.

  - **Métodos específicos:**
    - **obterNivel(int indice)** — Retorna um nível específico da fase pelo índice fornecido. Se o índice estiver fora do intervalo, exibe a mensagem de erro **"Index out of range."** e retorna **null**.

  - **Métodos Get e Set:** Métodos para acessar e modificar os valores dos atributos como:
    - **getNiveis() : ArrayList<Nivel>** — Retorna a lista de níveis da fase.

- **setNiveis(ArrayList<Nivel> niveis)** — Define a lista de níveis.

- **getExplicacao() : String** — Retorna a explicação da fase.

- **setExplicacao(String explicacao)** — Define a explicação da fase.

2.3 Classe NivelCarta

- **Propósito**

  Implementa níveis baseados em mecânicas de cartas.

- **Atributos**

  - **cartas : ArrayList<Carta>** — Conjunto de cartas no nível.
  - **dicas : ArrayList<String>** — Dicas disponíveis para o jogador.

- **Métodos**

  - **Construtor:** Inicializa todos os atributos da classe NivelCarta, incluindo a lista de cartas e as dicas disponíveis para o nível, garantindo que o objeto esteja pronto para uso com todos os dados necessários para a mecânica de cartas do jogo.

  - **Métodos específicos**

    - **obterCarta(int indice)** — Retorna uma carta específica do nível pelo índice fornecido. Se o índice estiver fora do intervalo (0 a tamanho da lista - 1), exibe a mensagem de erro **"Index out of range."** e retorna **null**.

    - **obterDica(int indice)** — Retorna uma dica específica da lista de dicas pelo índice fornecido. O método verifica se o índice está dentro do intervalo válido (0 a tamanho da lista - 1). Caso contrário, retorna null e exibe a mensagem de erro "Dica não encontrada."

  - **Métodos Get e Set:** Métodos para acessar e modificar os valores dos atributos como:
    - **getCartas() : ArrayList<Carta>** — Retorna a lista de cartas do nível.

- **setCartas(ArrayList<Carta> cartas)** — Define a lista de cartas do nível.

- **getDicas() : ArrayList<String>** — Retorna a lista de dicas disponíveis.

- **setDicas(ArrayList<String> dicas) —** Define a lista de dicas disponíveis no nível.

2.4 Classe Carta

- **Propósito**

  Define uma entidade de carta, com propriedades visuais e textuais.

- **Atributos**

  - **nome : String** — Nome da carta.
  - **id : int** — Identificador único da carta.
  - **imagemDispositivo : String** — Caminho para a imagem do dispositivo.
  - **descricao : String** — Texto explicativo associado à carta.

- **Métodos**

  - **Construtor:** Inicializa todos os atributos da classe, incluindo o identificador único (**id**), o nome da carta, o caminho da imagem no formato PNG (**imagemDispositivo**) e a descrição associada à carta.

  - **Métodos get e set:**

    - **getNome() : String** — Retorna o nome da carta.

    - **setNome(String nome)** — Define o nome da carta.

    - **getId() : int** — Retorna o identificador único da carta.

    - **setId(int id)** — Define o identificador único da carta.

    - **getImagemDispositivo()** : String — Retorna o caminho do arquivo de imagem no formato PNG associado ao dispositivo que será exibido na carta.

    - **setImagemDispositivo(String imagemDispositivo)** — Define o caminho do arquivo de imagem no formato PNG associado ao dispositivo que será exibido na carta.

    - **getDescricao() : String** — Retorna a descrição da carta.

- **setDescricao(String descricao)** — Define a descrição da carta.

2.4 Classe NivelContexto

- **Propósito**

Representa níveis baseados em contextos, envolvendo cenários e opções de resposta.

- **Atributos**

  - **cenario : Cenario** — Cenário do nível.
  - **opcoes : ArrayList<Carta>** — Opções de resposta disponíveis.
  - **idResposta : int** — Identificador da resposta correta.

- **Métodos**

  - **Construtor:** Inicializa os atributos da classe, incluindo o cenário (**cenario**), a lista de opções de resposta (**opcoes**) e o identificador da resposta correta (**idResposta**).

  - **Métodos específicos:**

    - **validarResposta(int idRespostaEscolhida)** — Verifica se a resposta selecionada pelo jogador é a correta. O método compara o identificador da resposta escolhida (**idRespostaEscolhida**) com o identificador da resposta correta (**idResposta**). Retorna true se os identificadores forem iguais e false caso sejam diferentes.

      Exemplo de comportamento:

      Entrada: idRespostaEscolhida (int) — O identificador da resposta escolhida pelo jogador.

      Processo: Compara idRespostaEscolhida == idResposta.

      Retorno:
      - true se os identificadores forem iguais.
      - false caso contrário.

  - **Métodos get e set:**

- **getCenario() : Cenario** — Retorna o cenário do nível.

- **setCenario(Cenario cenario)** — Define o cenário do nível.

- **getOpcoes() : ArrayList<Carta>** — Retorna as opções de resposta.

- **setOpcoes(ArrayList<Carta> opcoes)** — Define as opções de resposta.

- **getIdResposta() : int** — Retorna o identificador da resposta correta.

- **setIdResposta(int idResposta)** — Define o identificador da resposta correta.

2.5 Classe Cenario

- **Propósito**

  Define cenários específicos para níveis contextuais.

- **Atributos**
  - **pergunta : String** — Pergunta associada ao cenário.
  - **imagemCenario : String** — Caminho para a imagem do cenário.

- **Métodos**

  - **Construtor:** Inicializa os atributos da classe, incluindo a pergunta associada ao cenário (**pergunta**) e o caminho da imagem do cenário (**imagemCenario**).

  - **Métodos get e set:**

    - **getPergunta() : String** — Retorna a pergunta do cenário.

    - **setPergunta(String pergunta)** — Define a pergunta do cenário.

    - **getImagemCenario() : String** — Retorna o caminho da imagem.

    - **setImagemCenario(String imagemCenario)** — Define a imagem do cenário.

Diagrama de Classe