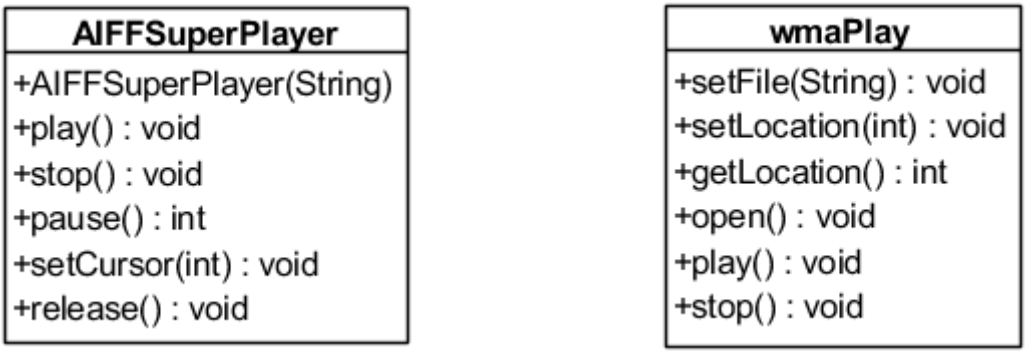
Unidade 2 – Problema 1

Sua equipe foi contratada para desenvolver uma biblioteca de classes para manipulação de áudio de forma padronizada.

Há disponíveis no mercado vários formatos de arquivo de áudio:

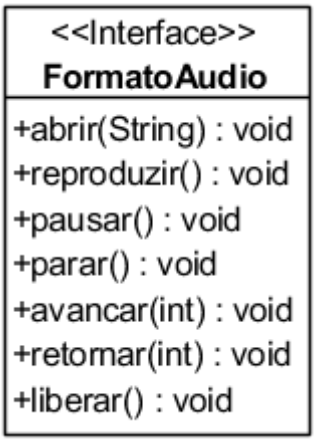
A empresa contratante está cansada de ter que trabalhar com diversas classes de formatos de áudio, em que cada uma possui sua própria interface. Por exemplo, observe abaixo as classes que manipulam os formatos AIFF e WMA:



As classes utilizadas pela empresa[1] estão disponíveis no AVA (Unidade 2/Problema 1) e são wmaPlay, AIFFSuperPlayer e WAVPlayer.

Assim, a empresa está solicitando que todas as classes sejam manipuladas de forma padrão, de acordo com a interface FormatoAudio.

[1] Estas classes não são reais, isto é, não realizam de fato a manipulação de arquivos de áudio. Estes componentes foram criados para fins didáticos. Com isso, tais componentes apenas simulam a operação de arquivos através de mensagens na console Java (System.out).



O método abrir(String) carrega um arquivo de áudio na memória, recebendo como parâmetro o nome do arquivo.

O método reproduzir() reproduz o arquivo de áudio que foi carregado na memória pelo método abrir(). A reprodução ocorre a partir do início do arquivo (na primeira vez) ou no ponto onde a reprodução tenha sido pausada.

O método pausar() para a reprodução do arquivo.

O método parar() para a reprodução e retorna para o início do arquivo.

O método avançar(int) recebe como parâmetro a quantidade de segundos que devem ser avançados no arquivo a partir do ponto atual.

O método retornar(int) recebe como parâmetro a quantidade de segundos que devem ser retrocedidos no arquivo a partir do ponto atual.

O método liberar() fecha o arquivo e libera a memória.

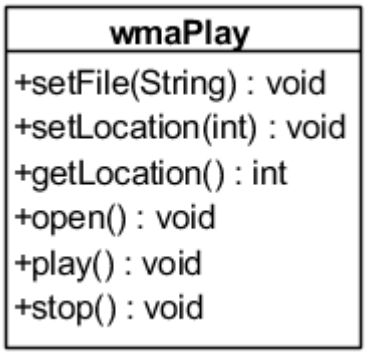
Além de querer trabalhar de forma padronizada através da interface FormatoAudio, a empresa também identificou que 70% dos seus desenvolvedores faz uso simplificado das classes de áudio. Então, ela também deseja que haja uma forma de trabalhar com as classes através de dois métodos:

reproduzirSimples(String): void – abre o arquivo especificado pelo parâmetro e o reproduz a partir do início.

pararSimples(): void – encerra a reprodução e libera a memória e o arquivo.

Para apresentar a solução para a empresa, a equipe deve criar um programa que utilize as classes wmaPlay, AIFFSuperPlayer e WAVPlayer através da interface FormatoAudio, fazendo a manipulação de arquivos de áudio de forma completa. E também demonstre o uso de forma simplificada.

Classes usadas pela empresa:



O método setFile() é utilizado para definir o nome do arquivo que será utilizado pelo objeto de reprodução de arquivos wma.

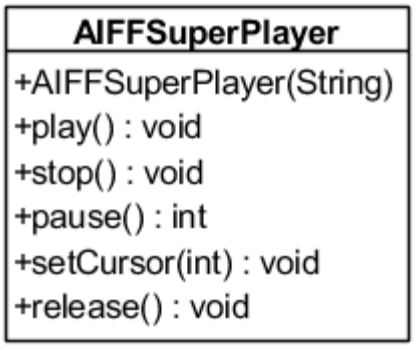
O método open() é utilizado para abrir o arquivo definido pelo método setFile(). Porém, não define qual é o ponto inicial de reprodução.

O método setLocation() é utilizado para indicar a posição do arquivo (segundos) onde deve iniciar a reprodução. Para começar a partir do início deve-se fornecer como argumento o valor "0".

O método getLocation() retorna em que posição (segundos) se encontra a reprodução do arquivo.

O método play() reproduz o arquivo aberto com o método open(). O arquivo de áudio começa a ser reproduzido a partir da posição indicada pelo método setLocation.

O método stop() para a reprodução do arquivo. Caso a próxima mensagem seja play(), reinicia a execução do ponto onde parou. Caso seja stop(), volta ao início do arquivo (setLocation(0)).



O construtor recebe o nome do arquivo a ser aberto e o abre, deixando-o pronto para execução.

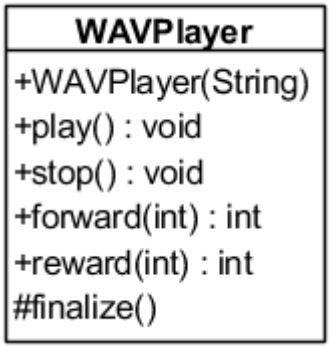
O método play() reproduz o arquivo aberto, a partir do ponto em que se encontra.

O método pause() para a reprodução do arquivo, retornando a posição (segundos) em que se encontra.

O método stop() para definitivamente a reprodução do arquivo. Não pode ser retomado por play() sem que antes seja definida uma posição (setCursor).

O método setCursor(int) define uma nova posição (segundos) do arquivo. Pode estar em reprodução ou parado.

O método release() libera o arquivo da memória. Caso o arquivo esteja em reprodução, ele primeiro para (stop).



O construtor recebe o nome do arquivo a ser aberto e o abre, deixando-o pronto para execução.

O método play() reproduz o arquivo aberto, a partir do ponto em que se encontra.

O método stop() para a reprodução do arquivo. Pode ser retomado do ponto em que parou com play().

O método forward(int) recebe como parâmetro um valor em milissegundos em que deve ocorrer um salto para frente na posição do arquivo. Seu retorno é a nova posição, também em milissegundos.

O método reward(int) recebe como parâmetro um valor em milissegundos em que deve ocorrer um salto para trás na posição do arquivo. Seu retorno é a nova posição, também em milissegundos.

O método finalize() é o destrutor, em que o arquivo é fechado e a memória liberada.