1. Capa: identificando o curso, o tema, a relação de alunos do grupo (nome/RA)

UNIVERSIDADE PAULISTA

CURSO DE CIÊNCIA DA COMPUTAÇÃO & SISTEMAS DA INFORMAÇÃO

APS – ATIVIDADES PRÁTICAS SUPERVISIONADAS

SANTOS – SP

2014

COLABORADORES

MATHEUS RODRIGUES MARTINS PEREIRA – RA: B73ABD-5 – Turma: CC4Q41

MATHEUS FELIPE DOS PASSOS E PAZ – RA: B57IAJ-0 – Turma: CC4Q41

VICTOR BRUNO DOS SANTOS PAIVA – RA: T13588-5 – Turma: SI3P41

LEONARDO PEREIRA MOREIRA – RA: A96730-3 – Turma: CC4Q41

APS – ATIVIDADES PRÁTICAS SUPERVISIONADAS

"DESENVOLVIMENTO DE UM JOGO COM UTILIZAÇÃO DE INTERFACE GRÁFICA"

Atividades Práticas Supervisionadas trabalho apresentado como exigência para avaliação do segundo bimestre, em disciplina do 4º semestre do curso de Ciência da Computação & Sistemas da Informação da Universidade Paulista, Sob orientação do professor Sergio Medina.

SANTOS – SP

2014

2. Índice

Índice

3. Objetivo e motivação do trabalho

Objetivo e motivação do trabalho

Reciclagem hoje é um tema muito abordado no mundo. Afinal todos nós estamos tentando ajudar o meio ambiente a sobreviver com toda essa poluição, fazendo com que lixos, gases entre outros poluentes sejam diminuídos. A motivação do trabalho é levar a reciclagem para os jogadores, fazendo com que a diminuição de lixo comece em casa.

O tema do trabalho dado é sobre a reciclagem de lixo considerando uma metrópole, ou seja, uma cidade hoje produz toneladas de lixo por dia e uma grande parte desse lixo pode ser reaproveitada pelas empresas que fabricam o material ou até mesmo pelos moradores, o lixo hoje é um dos grandes poluentes no mundo.

A nossa aplicação foi criada com uma base de um jogo de console e se chama Guitar Hero, onde a musica começa a tocar e você fica responsável pelas notas da guitarra, e conforme a nota aparece na tela, deve ser pressionado o botão correspondente. No jogo criado o sistema foi quase o mesmo, porém, não toca nenhuma musica, apenas ficam as lixeiras com as letras de teclado correspondentes, e aparece uma imagem com o nome do lixo e o jogador deve pressionar o botão correspondente ao lixo que pode se jogar.

O nosso jogo tem como objetivo a ensinar as pessoas sobre quais os materiais podem ou não ser reciclados, onde as pessoas podem botar o jogo em pratica na própria casa diminuindo assim a quantidade de lixo mandada para os esgotos, rios, mares, terrenos que danificam o meio ambiente.

4. Introdução

INTRODUÇÃO

Reciclagem

É a metodologia que tem como objetivo reutilizar materiais usados e resíduos como ingredientes essenciais na fabricação de outros produtos. Desta forma, estas substâncias são novamente inseridas no circuito produtor de onde elas mesmas se originaram.

Muitos materiais podem ser reciclados e os exemplos mais comuns são o papel, o vidro, o metal e o plástico. As maiores vantagens da reciclagem são a minimização da utilização de fontes naturais (muitas vezes não renováveis) e a minimização da quantidade de resíduos que necessita de tratamento final, como aterramento ou incineração.

O conceito de reciclagem serve apenas para os materiais que podem voltar ao estado original e ser transformado novamente em um produto igual em todas as suas características. O conceito de reciclagem é diferente do conceito de reutilização.

Vantagens da reciclagem

1 - A diminuição do consumo de matérias-primas virgens (muitas delas não são renováveis e podem apresentar ainda exploração dispendiosa).

2 - Contribui para diminuir a poluição do solo, água e ar.

3 - Melhora a limpeza da cidade e a qualidade de vida da população.

4 - Prolonga a vida útil de aterros sanitários.

5 - Melhora a produção de compostos orgânicos.

6 - Gera empregos para a população não qualificada e receita para o pequeno e micro empresário.

7 - Gera receita com a comercialização dos recicláveis.

8 - Estimula a concorrência, uma vez que os produtos gerados a partir dos reciclados são comercializados em paralelo a aqueles gerados a partir de matérias-primas virgens.

9 - Contribui para a valorização da limpeza pública e para formar uma consciência ecológica.

Importância do projeto para o tema

A importância do projeto feito para a reciclagem, é a conscientização de uma forma mais agradável, através de um jogo. Onde se pode obter o conhecimento dos tipos de lixos que são reciclaveis e também dos objetos que devem ser depositados neles.

Tipos de lixo

5. Regras do jogo (conceitos gerais)

REGRAS DO JOGO (CONCEITOS GERAIS)

Conforme o objeto aparece na tela, o jogador deve clicar ou pressionar no teclado o botão do lixo correspondente, enquanto o jogador pensa, uma barra de tempo vai diminuindo e o jogador não pode deixar que essa barra chegue ao fim se não o jogo acaba.

Conforme o jogador coloca os objetos nos lixos certos, a barra de tempo aumenta e o mesmo ganha 5 pontos, porém, se ele errar o lixo, a barra diminui mais ainda e o jogador perde 2 pontos.

O jogo acaba quando o jogador termina de responder a todos os objetos, para ganhar ele deve somar os pontos necessários conforme a dificuldade escolhida, e também não pode deixar a barra de tempo chegar ao fim.

Quanto maior a dificuldade escolhida, mais pontos o jogador deve fazer para vencer e mais variedades de lixos e objetos irão aparecer.

Para descobrir quantos pontos o jogador precisa fazer para vencer o jogo e a diferença entre as dificuldades, ele deve clicar no botão ‘sobre’ localizado na mesma janela onde se escolhe o nível.

6. Plano de desenvolvimento do jogo (elementos e ferramentas que serão utilizadas).

Plano de Desenvolvimento

O jogo contem 1 pacote com 5 classes e 3 folders com um banco de imagens utilizadas no próprio jogo.

A princípio o jogo tem como objetivo e tema a reciclagem, então, foi desenvolvido um jogo no qual você precisa identificar o lixo e logo após clicar ou pressionar o botão correspondente a lixeira correspondente ao lixo. O jogo possui um menu inicial que você pode tanto quanto prosseguir quanto sair do jogo, logo após caso prossiga, você se depara com a tela de dicas do jogo e seleção de nível, você pode tanto voltar para o menu principal quanto ir para a tela do jogo. Na tela do jogo, você verá as lixeiras desabilitadas esperando que o jogador comece para que sejam habilitadas, quando começado, o jogo contabiliza os pontos, erros, acertos, combos e quantas palavras e/ou lixo foram mostrados na tela, tudo isso durante a jogatina, você tem o objetivo de marcar “x” pontos de acordo com os níveis.

A primeira classe que foi criada foi Jogo.java, porque precisávamos de uma base, e a base do jogo era o próprio jogo em si.

As lixeiras, os botões, a barra de progresso, os contadores de pontuação e os combos foram importados tanto Swing quanto AWT.

Para que as 90 palavras fossem criadas e colocadas em suas respectivas lixeiras, fizemos vetores de 0 à 89 para cada lixo, em um jogo é impossível um lixo se repetir.

Um passo importante para o desenvolvimento foi a interação da classe Jogo com a classe Opcoes. Na linha de código a seguir, estamos recebendo o valor retornado do método getA(), que no caso é o valor escolhido na comboBox cbNivel da classe Opcoes:

```
int nivel = Opcoes.getA();
```

Feito isso, temos o nível escolhido pelo usuário em mãos, sendo 0 para o nível fácil, 1 para o médio e 2 para o difícil. A diferença entre eles é o número de palavras e a pontuação mínima, portanto temos esse trecho do código fonte:

```
if (nivel == 0)
                {
                        npalavras = 60;
                        titulo = "Fácil";
                        pontMinima = 230;
                }
                else if (nivel == 1)
                {
                        npalavras = 71;
                        titulo = "Médio";
                        pontMinima = 280;
                }
                else if (nivel == 2)
                {
                        npalavras = 90;
                        titulo = "Difícil";
                        pontMinima = 330;
                }
```

A variável titulo mostrada acima é uma variável que irá ser usada para mostrar a dificuldade no titulo do frame, portanto nada que interfira diretamente na forma de jogar.

O método sorteio é essencial para que o jogo funcione de forma correta. Nele é contido a parte do código que faz o sorteio de uma posição no vetor e atribui esse valor sorteado para a variável contPalavras. Usando esse número sorteado, o programa procura no vetor palavras qual é a palavra correspondente a aquela posição no vetor. E ainda depois o programa procura a imagem correspondente a esse número, que está associado a palavra escrita no vetor. Vamos mostrar a linha de código e dar um exemplo:

```
public void sorteio(){


        cont += 1;

        lblContagem.setText(String.valueOf(cont));

        String palavras [] = {/*Papel*/"Jornal", "Sulfite", "Folha de caderno",
"Envelope", "Caixa de pizza", "Cartolina", "Papel Cartão", "Revista", "Lista
telefônica", "Formulário", "Papel de Fax", "Fotocópia", "Caixa de Chá", "Caixa de
Lápis Vazia", "Caixa de remédio",
                                        /*Plastico*/"Garrafa Pet", "Sacola de
mercado", "Cano PVC", "Tampa de caneta", "Garrafa de óleo", "Copo descartável",
"Talher de festa", "Frasco de produto", "Embalagem de shampoo", "Embalagem de
detergente", "Galão de água", "CD", "Pratos de festa", "Saco de Lixo", "Bolsa de soro
medicinal",
                                        /*Vidro*/"Embalagem de Azeitona",
"Frasco de remédio","Copo", "Caco de vidro", "Tampa de forno", "Garrafa de
cerveja", "Garrafa de Refrigerante", "Frasco de perfume", "Embalagem de molho",
"Prato", "Tampa de microondas", "Garrafa de suco", "Vidro de janela", "Para-brisas",
"Tigela de vidro",
                                        /*Metal*/"Lata de refrigerante",
"Panela sem cabo", "Cano", "Tampinha de garrafa", "Arame", "Cobre", "Prego",
```

```
"Embalagem de Marmitex", "Papel Alumínio Limpo", "Ferragem", "Lata de sardinha",
"Tampa de iogurte", "Embalagem de café", "Lata de Creme de leite", "Lata de leite
condensado",
                                        /*Organico*/"Folhas", "Sementes",
"Ossos", "Restos de carne", "Palito de dente", "Borra de café", "Chiclete", "Resto de
bebidas", "Papel Molhado", "Fezes", "Urina",
                                        /*Não Reciclavel*/"Papel Higiênico",
"Espelho", "Clipes", "Interruptor", "Cinzeiro","Guardanapo", "Grampo", "Lata de
verniz", "Tubo de TV", "Lente de óculos", "Adesivo", "Etiqueta", "Papel Carbono",
"Papel Celofane", "Telefone", "Caneca", "Fotografia", "Bituca de cigarro", "Bandeja
de Plastico"};// O vetor que recebe as palavras mostradas para o usuário.

        contPalavras =(int) (Math.random() * npalavras);//Sorteia um número
inteiro, este número será usado como índice do vetor palavras[]

    if (cont == npalavras)
        {
            i = 0;
        }

            if(!lista.contains(contPalavras))
            {
                lista.add(contPalavras);
            }
            else if (lista.contains(contPalavras))
            {
                while(lista.contains(contPalavras))
                {
                    if(cont < npalavras)
                        contPalavras =(int) (Math.random() *
npalavras);
                    else
                        break;
                }
```

```
                    lista.add(contPalavras);
                }
                if(cont < npalavras)
                {
                    lblPalavra.setText(palavras[contPalavras]);  // A
label lblPalavra recebe o valor do vetor de índice randomico.
                    lblPalavra.setIcon(new
ImageIcon(Jogo.class.getResource("/PCT_APS/Imagens/" + contPalavras +
".png")));
                }
        }
```

Este é o método sorteio. Vamos supor que o valor sorteado seja o valor 18. O programa vai buscar no vetor o índice de número 18, e mostra essa palavra pro usuário, que no caso seria “Tampa de caneta”. Depois disso, o programa busca na pasta Imagens localizada dentro do pacote, a imagem correspondente a essa palavra. A imagem vai estar nomeada com o número da posição dela no vetor e a extensão .png. Nesse caso, a imagem da tampa de caneta está nomeada como 18.png.

Fora isso vemos a utilização de listas dentro desse método, que serve para verificar se aquela posição no vetor já foi sorteada. Se foi, é feito um outro sorteio até que o programa sorteie um número que não está na lista, ou seja, que nunca foi utilizado.

Além disso temos um contador para verificar o número de vezes que esse método foi executado. Cada vez que ele é executado acrescenta 1. Isso serve para verificar quando que o jogo deve parar de sortear palavras, no caso, até a variável cont for igual a variável npalavras (variável que representa o número de palavras dependendo do nível escolhido). Quando isso acontecer, a variável i recebe o valor de 0. A variável i é a variável que controla a barra de progresso, e ela chegando a 0 o jogo chega ao fim.

A variável contPalavra faz com que todos os vetores fiquem aleatórios para que haja a dinâmica do jogo. Cada botão foi configurado um “IF” e “ELSE” para receber os lixos correspondentes, porém, como vetores. Exemplo:

```
if((contPalavras >= 0) && (contPalavras <= 14))//Se o vetor estiver entre as posições informadas, executa o código abaixo entre as chaves:
                {
                        acerto();
                }

                else//caso o índice do vetor não esteja entre as posições informadas acima, executa o código abaixo entre as chaves:
                {
                        erro();
                }
```

Os métodos acerto e erro foram criados para evitar uma repetição de código desnecessária, segue o código dos métodos:

```
public void acerto(){
        i += 10; //adiciona 10 à variavel i, ou seja, o usuário ganha mais tempo
        sorteio();//chama o método sorteio para chamar outro índice do vetor
        pbVelocidade -= 2;//aumenta a velocidade da barra de progresso
        lblResposta.setText("");
        acerto += 1;
        combo += 1;
        if (combo > 3)
        {
                pontuacao += 7;//acrescenta 7 pontos para a pontuação do usuário
        }
        else
        {
                pontuacao += 5;//acrescenta 5 pontos para a pontuação do usuário
```

```
                }
                lblCerto.setText(String.valueOf(acerto));
                contCombo.setText(String.valueOf(combo));
                lblResposta.setIcon(new
ImageIcon(Jogo.class.getResource("/PCT_APS/Logos/acertou.png")));
                if(combo >= 3)
                {
                lblCombo.setVisible(true);
                lblCombo.setIcon(new
ImageIcon(Jogo.class.getResource("/PCT_APS/Logos/combo.png")));
                contCombo.setVisible(true);
                }
        }

        public void erro(){
                i -= 10;//decrementa 10 aà variavel i, ou seja, o usuário perde tempo
                sorteio();//chama o método sorteio para chamar outro índice do vetor
                pontuacao -= 2;//decrementa 2 pontos para a pontuação do usuário
                lblResposta.setText("");
                erro += 1;
                combo = 0;
                contCombo.setText("0");
                contCombo.setVisible(false);
                lblErrado.setText(String.valueOf(erro));
                lblResposta.setIcon(new
ImageIcon(Jogo.class.getResource("/PCT_APS/Logos/errou.png")));
                lblCombo.setVisible(false);

        }
```

Percebe-se que no método acerto ele verifica se o usuário está dentro de um combo, se estiver acrescenta 7 para a pontuação, se não ele acrescenta 5. Fora isso ele chama novamente o método sorteio, para realizar o sorteio de uma nova palavra

e imagem, e ainda aumenta um pouco a barra de rolagem através da linha i+=10;. Um segredo que não é mostrado para o usuário é que, dentro desse método acerto existe uma linha de código que é pbVelocidade -= 2;, isso significa que a cada acerto do usuário, a barra de progresso se decrementa cada vez mais rápido.

No método erro é o contrário do método acerto, ele tira 2 pontos variável pontuação, diminui mais um pouco a barra de progresso e, da mesma forma que o método acerto, também chama o método sorteio para realizar um sorteio de uma nova palavra.

Notar que a label lblResposta é quem recebe a imagem em forma de ícone da informação que aparece no canto do jogo, se o usuário acertou ou se errou.

As lixeiras são JButton's no qual foram colocados ícones das lixeiras respectivas, a principio, os as lixeiras só obtinham o clique do mouse, mas para ficar mais dinâmico foi configurado botões no teclado para cada lixeira, os botões são: Q = Papel, W = Plástico, E = Vidro, R = Metal, T = Orgânico e Y = Não Reciclável. Para que conseguíssemos isso, foi preciso usar KeyStroke, KeyEvent, InputMap e ActionMap. Segue a lógica:

KeyStroke keyStroke = KeyStroke.*getKeyStroke*(KeyEvent.*VK_TECLA CORRESPONDENTE NO TECLADO*, 0);

String actionName = "TECLA_TECLA CORRESPONDENTE NO TECLADO";

InputMap inputMap = btnNOME DA LIXEIRA CORRESPONDENTE.getInputMap(JComponent.*WHEN_IN_FOCUSED_WINDOW*);

inputMap.put(keyStroke, actionName);

ActionMap actionMap = btnLIXEIRA CORRESPONDENTE.getActionMap();

actionMap.put(actionName, nemonicoNOME DA LIXEIRA CORRESPONDENTE);

Foi criado o objeto keyStroke da Classe KeyStroke, nele é mapeado o botão correspondente no teclado. InputMap direciona o botão que deve ser utilizado. ActionMap faz a ligação do botão com a tecla do teclado. Deixando assim, o jogo mais dimamico.

E quando a variável i chega a 0, o programa verifica se você fez os pontos necessários e se você conseguiu responder a todas as palavras. Caso sim, a seguinte linha de comando é chamada.

```
if ((cont == npalavras)&&(pontuacao >= pontMinima))
        {



                try {
                        new Vencedor().setVisible(true);
                } catch (IOException e) {

                        e.printStackTrace();          }
                dispose();
        }
```

Basicamente, ele está chamando a classe Vencedor que é a classe que apresenta uma mensagem parabenizando ao jogador, e logo em seguida fecha o frame do jogo.

Porém caso você não tenha conseguido os pontos necessários ou não tenha conseguido responder a todas as palavras, o programa irá verificar o motivo da sua derrota e irá lhe informar.

```
{
        if((cont < npalavras)&&(pontuacao >= pontMinima))
        {
                fim = "Você fez os pontos necessários mas o tempo acabou!\n\nVocê
        perdeu!";
    }
         else if((cont == npalavras)&&(pontuacao < pontMinima))
        {
                fim = "Você respondeu todas as palavras dentro do tempo mas não fez
os pontos necessários\n\nVocê perdeu!";
```

```
        }
         else if((cont < npalavras)&&(pontuacao < pontMinima))
        {
                fim = "O tempo acabou e você não fez os pontos necessários.\n\nVocê perdeu!";
        }

        int resposta = JOptionPane.showConfirmDialog(null, fim + "\n\nPontuação: " + pontuacao + " pontos\n\nDeseja jogar novamente?", "Fim de jogo!", JOptionPane.YES_NO_OPTION);//mostra uma caixa de mensagem no final, mostrando a pontuação final
        if(resposta == JOptionPane.YES_OPTION)
        {
        try {
                        Jogo frame = new Jogo();
                                                                frame.setVisible(true);
                } catch (Exception e) {
                                                                e.printStackTrace();
                }
                dispose();

        }
```

Isso encerra o jogo da seguinte maneira, em caso de derrota, o jogo vai mostrar o motivo e sua pontuação através de uma caixa de mensagem, e nessa mesma caixa lhe oferecendo a jogar novamente. O motivo da derrota é controlado através da variável fim, do tipo String.

7. Projeto (estrutura e módulos que serão desenvolvidos) do programa

Projeto

No começo, fizemos uma reunião para decidir o que fazer, qual jogo. Dentre várias opiniões, decidimos fazer um jogo que fosse dinâmico, no qual o jogador pudesse interagir com o jogo.

Foi feito vários desenhos em caderno para chega em uma conclusão de como seria a interface gráfica até que todos chegaram em um consenso e foi decido.

O jogo tem como base a reciclagem de lixo, que consiste em ver o lixo que aparece na tela e jogar na lixeira correspondente.

A classe Jogo.java foi a primeira classe a ser criada, nela foi criado a interface gráfica que planejávamos, com 6 lixeiras, com barra de tempo e botão para iniciar o jogo.

Dividimos as tarefas entre os 4 integrantes, sempre sincronizando os projetos.

Assim que foi criado a primeira parte, o jogo mostrava o nome do lixo e tínhamos que clicar no botão (que era a lixeira) para que pudéssemos ter um acerto ou um erro. O acerto e erro impactava no tempo.

Cada vez que o projeto era aberto e a cada reunião, vinham propostas de todos nós para que fossem feitas melhorias.

Colocamos em uma segunda etapa o contador de pontos e palavras, assim, o jogador sabia quantos pontos ele ia ganhando e quantas palavras já foram, até que decidimos criar o Menu inicial ou tela de inicio.

A criação da tela de inicio foi um tanto quanto trabalhoso, porque necessitava de uma arte para ser colocada e decidir um nome para o jogo. Foram várias sugestões para o nome do jogo, mas nenhuma parecia se encaixar no nosso projeto, até que um nome foi decidido: Reciclando com Pendoleo.

A nome Reciclando com Pendoleo foi decidido pelo grupo e faz alusão ao nome do Pen Drive do integrante Leonardo Pereira. Com o nome decidido, partimos para os desenhos de como deveria ser a tela inicial do jogo.

A idéia é que mostrasse o objetivo do jogo logo na tela inicial, que era reciclagem. Decidimos então colocar o símbolo de reciclagem junto ao nome do jogo. Como o eclipse não ofereceu recursos necessários para a criação da arte e do nome do jogo como queríamos, foi necessário o uso da ferramenta Photoshop.

No Photoshop foi criado o símbolo de reciclagem em coloração verde, para lembrar o meio ambiente, e o nome do jogo. No nome do jogo foi acrescentado no lugar do “L” um pendrive em forma de “L” para que ficasse claro a intenção do nome do Pen Drive.

Com as artes criadas, criamos a classe Menu.java. No Menu.java, colocamos o botão para iniciar o jogo, que no caso te transferia para tela do jogo e o botão “Sair”.

No meio da tela foi a hora de colocar a arte criada anteriormente, o logo de reciclagem verde por traz e o nome do jogo na frente dele. O nome do jogo foi colocado em cor azul para representar o Pen Drive que era também azul. Para ficar mais claro ainda, colocamos no canto superior esquerdo o desenho do Pen Drive que é o “Pendoleo”.

Decidimos também que o fundo da tela seria verde água, também para lembrar o meio ambiente.

Parecia que faltava algo quando transitava-se entre a tela de Menu inicial e a tela do jogo. Foi quando decidimos criar uma tela que intermediasse, a tela Opções.

Na tela de opções, havíamos pensado em colocar somente a seleção de níveis, que são: Fácil, Médio e Difícil e foi o que aconteceu. Porém, ainda não tínhamos pensado como que seria a diferença entre os níveis.

O jogo estava particularmente enjoativo, você obtinha pontos e tempo adicional para cada acerto e não tinha fim, fazendo você fechar o jogo quando cansasse. Juntando as duas situações resolvemos conversar sobre a implementação dos níveis.

Estávamos em duvida se os níveis implicariam em mais palavras, quanto mais difícil mais palavras, ou se quanto mais difíceis, mais lixeiras e meta de pontos.

Se colocássemos mais palavras de acordo com a dificuldade, havia a probabilidade o jogo ficar muito fácil nos níveis mais fáceis e talvez até enjoativo novamente no nível difícil. O plano B de colocar mais lixeiras quanto mais difíceis foi a escolhida, porque se você tem uma meta de ponto com uma quantidade X de lixos você tem uma motivação de tentar bater a meta em menor tempo possível.

Então foi decidido que:

No nível fácil o jogo viria com 4 lixeiras, para bater a meta você precisa de no mínimo 230 pontos e tem um limite de 60 palavras para tentar atingir a meta.

No nível médio o jogo viria com 5 lixeiras, para bater a meta você precisa de no mínimo 280 pontos e tem um limite de 71 palavras para tentar atingir a meta.

No ultimo nível, o difícil, o jogo viria com todas as 6 lixeiras, para bater a meta você precisa de no mínimo 330 pontos e tem um limite de 90 lixos para tentar atingir a meta.

Agora o jogador poderia começar com o básico e ir melhorando e dificultando o jogo gradativamente com o seu aprendizado.

Depois de criada as três classes principais, o jogo parecia estar pronto, mas parecia que o grupo sempre queria melhorar o jogo em todos os sentidos para que pudesse ser um jogo impecável na medida do possível.

Até que em mais uma reunião em grupo, foi decido colocar na tela de opções, as dicas do jogo. As dicas do jogo foi implementada para que o jogador possa conhecer toda a dinâmica do jogo e ser guiado para não estranhar o jogo durante o mesmo. Nela tem o objetivo do jogo, as regras e um explicativo para os níveis que correspondia ao botão “Sobre”.

O botão “Sobre” trazia o resultado de uma nova classe criada, a classe Pontuação.Java.

Em Pontuação. Java, colocamos as informações dos níveis, como foi descrito anteriormente.

O jogo parecia completo, mas se o jogador conseguisse atingir a meta, o que ele ganharia?

Criamos uma nova classe chamada Vencedor.Java que era responsável por retornar para o jogador assim que ele batesse a meta uma mensagem: “Parabéns! Você ganhou! Você fez “X”(depende do nível o número necessário de pontos) pontos e um boneco segurando um troféu, no final, em baixo, há uma mensagem perguntando se deseja jogar novamente, com um 2 botões, de sim e de não.

Caso jogador não bata a meta, aparece uma mensagem na tela:

O tempo acabou e você não fez os pontos necessários.

Você perdeu!

Pontuação “X”

E também há uma mensagem perguntando se deseja jogar novamente, com 2 botões, de sim e de não.

A ultima coisa adicionada no jogo foi o contador de combo, decidimos que se o jogador adquire uma experiência e acerta vários lixos em suas lixeiras seguidamente, ele merece um aditivo na pontuação, então se o jogador começa a fazer combo (que é acertar varias seguidas) ele possui um adicional no ponto e na tela mostra o quanto de combo ele já está. Caso o jogado erre, o contador de combo é zerado e caso ele volte a acertar e fazer combo o contador começa a ser contado a partir do 1, porém o combo só é mostrado na tela a partir do 3. Toda vez também que o jogador acerta aparece na tela uma palavra dizendo: Acertou, e caso o jogador erre aparece também na tela a mensagem dizendo: errou.

Depois do jogo testado varias vezes e percebendo que não havia mais nada que pudéssemos adicionar, resolvemos finalizar o projeto que se tornou o nosso mais novo jogo chamado “Reciclando com Pendoleo”.

8. Relatório com as linhas de código do programa

Devido ao código fonte ter ficado muito extenso, o mesmo foi postado em um arquivo pdf separado.

9. Apresentação do programa em funcionamento em um computador, apresentando todas as funcionalidades pedidas e extras.

Ao executar o jogo, ele abre primeiramente a interface da classe Menu (classe que contém o método main()). Este é o menu principal do jogo, onde você tem apenas duas opções: Sair, onde o programa chama System.exit(0); e fecha a aplicação na hora, ou a opção Iniciar, que chama a classe Opcoes.java.

No menu também fica claro o propósito do jogo e o tema que foi escolhido para abordagem no trabalho, a reciclagem.

Todos as edições como recorte de fotos, transparência, foram feitos utilizando o Adobe Photoshop. As imagens do menu, que estão salvas nas pastas localizadas dentro do próprio projeto, foram editadas e salvas em .png, para poder utilizar o fundo transparente. Aliás, todas as imagens do jogo foram salvas nesse formato para uma melhor visualização do usuário e também por questão de aparência.

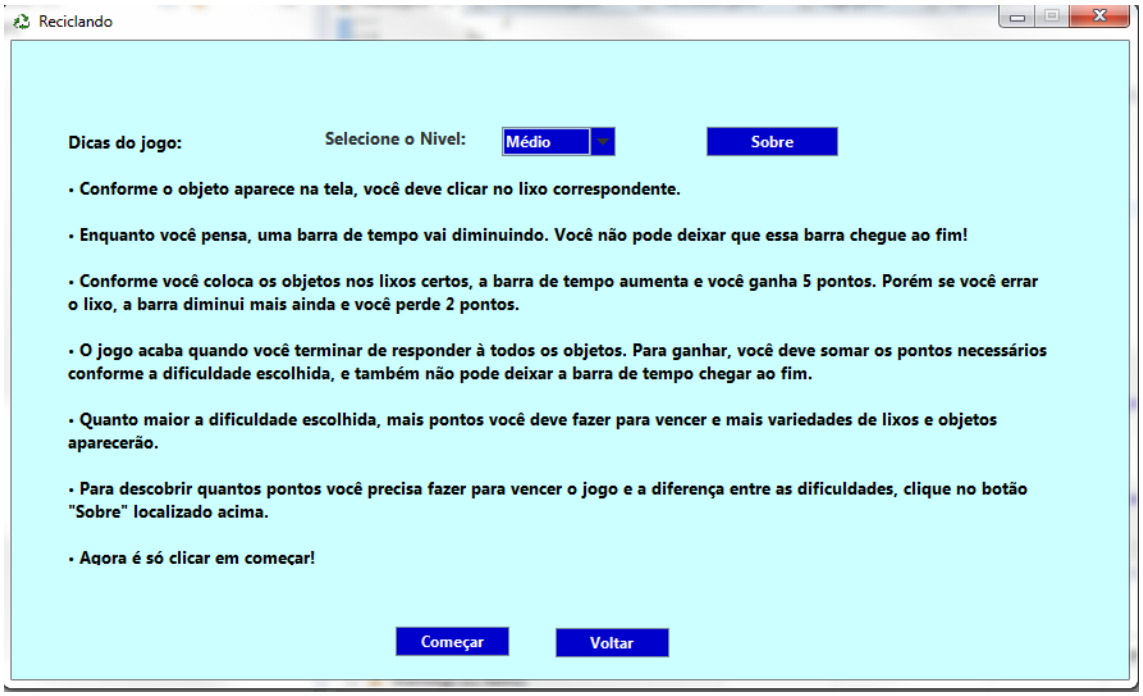
Abaixo segue o *print screen* da tela do menu principal, onde dá pra ver como que o trabalho foi feito, imagem por imagem, sem deixar escapar nenhum detalhe.

Passando para o próxima tela do jogo temos a interface da classe Opcoes.java. Essa classe foi criada especificamente para que o usuário possa escolher o nível que deseja jogar, e também para ele conhecer um pouco melhor as regras do jogo.

Aqui você vê detalhadamente como que o jogo funciona, e decide como é a forma mais fácil de vencer. Para ir treinando e ver como o jogo funciona, é recomendado começar pelo nível fácil.

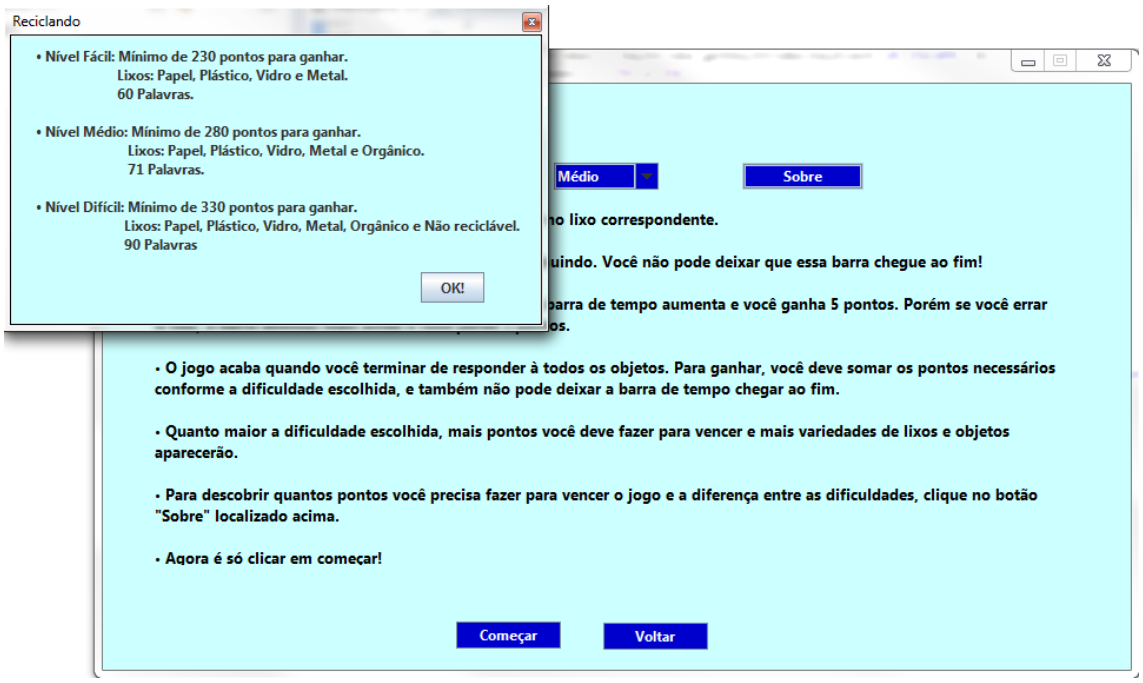
Nessa tela você também encontra, além das dicas e da escolha do nível, a opção “Sobre” (que será mostrada mais a frente), a opção “Começar” que dá acesso a classe Jogo.java, e a opção voltar que retorna para a classe Menu.java, que é a classe do menu principal mostrada anteriormente.

Nessa tela dá pra começar a notar que foi estabelecido uma cor padrão para todas as interfaces de cada classe, uma cor padrão para todos os botões do jogo, e também um ícone padrão para todos os frames. Abaixo segue o print-screen da interface da classe Opcoes.java.

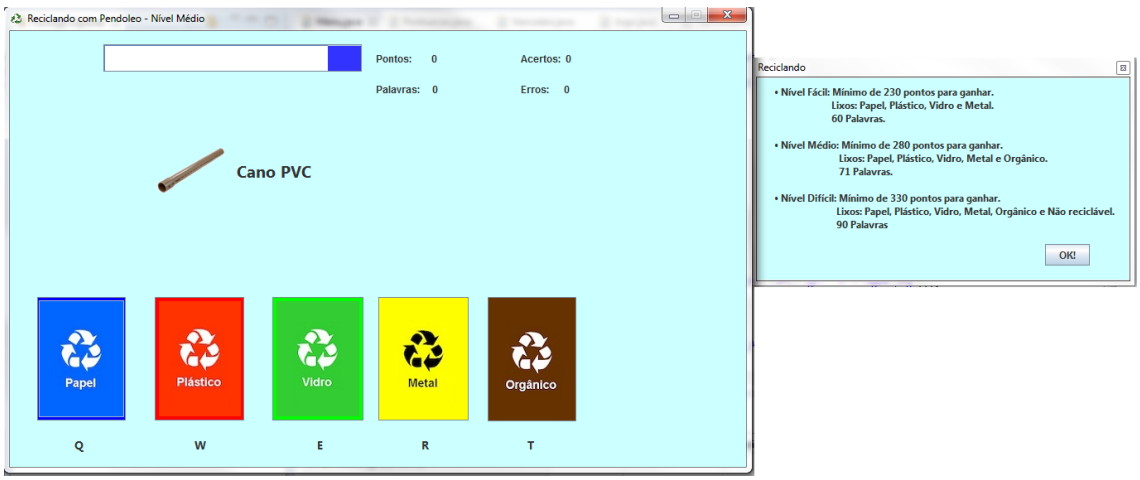


Ainda na tela de Opcoes.java, existe ali ao lado dos níveis uma opção chamada “Sobre”. Ali está uma orientação para o usuário saber quantos pontos precisa saber, quais são as lixeiras que ele vai ter que acertar, e ainda quantas palavras que aparecerão para ele.

Foi criada uma classe para mostrar essas informações, por questão de aparência e design do jogo, visto que poderia ter sido criado uma caixa de mensagem para tal ação. Veja como ficou:



O usuário também tem a possibilidade de começar o jogo com essa janela aberta para fins de informação, sem precisar voltar para as opções. Para isso basta ele começar o jogo e mover a tela das informações para o lado, de forma que ela fique em segundo plano e sem atrapalhar o jogo. Veja abaixo o print-screen da tela do Jogo com a tela Sobre em segundo plano.



Até aqui, mostramos apenas os recursos da parte pré-jogo da aplicação. Porém, as maiores funcionalidades estão contidas na classe Jogo.java. Essa classe contém todo o código do Jogo, carrega todas as imagens e etc.

Mas vamos começar pelo princípio de tudo. Ao abrir a classe jogo, o usuário se depara com uma tela onde aparece apenas o botão “Iniciar”, que inicia o jogo na hora, o botão “Voltar”, que volta para a classe Opcoes, uma barra de progresso vazia e os botões das lixeiras localizados abaixo. Estes botões estão desativados, e só são ativos no momento que é clicado o botão “Iniciar”. Abaixo de cada lixo estão as teclas de atalho, que são teclas que facilitam a jogabilidade pelo fato de tornar a sua resposta mais rápida. Foram escolhidas as teclas QWERTY por serem próximas e ao nosso ver, as mais utilizadas em jogos populares. Caso prefira, pelo clique do mouse também é possível escolher a sua resposta. Perceba também que o nível que você escolheu na tela anterior irá aparecer no título do frame.

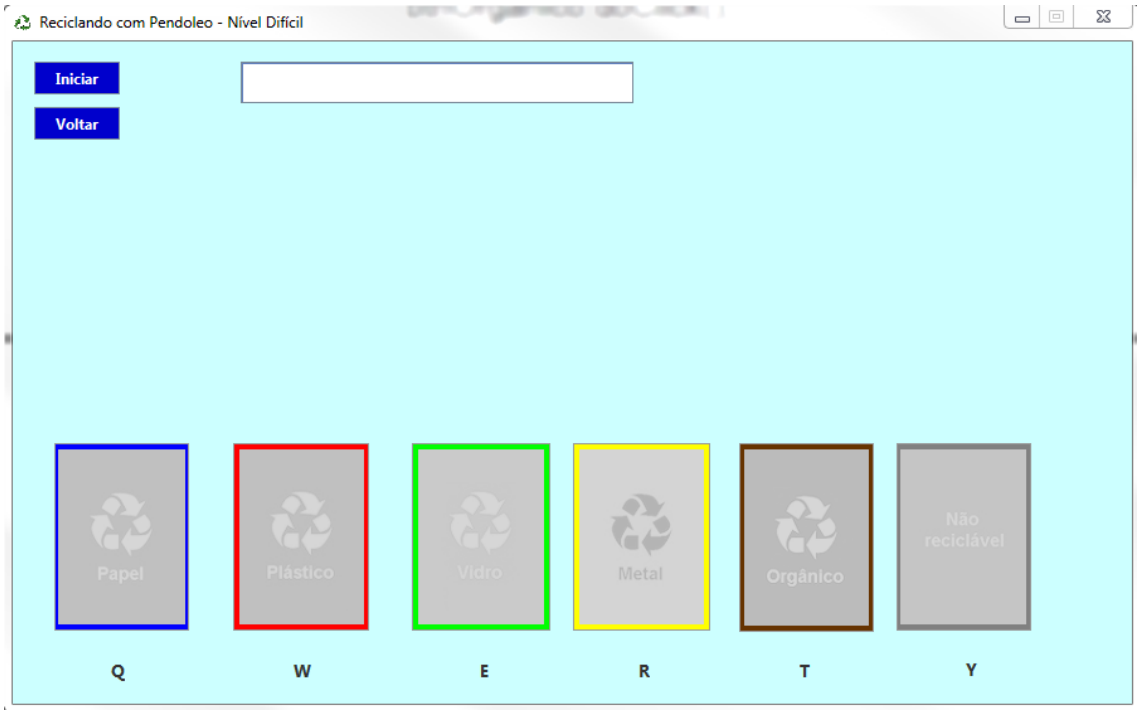
As telas que serão mostradas são referentes ao nível difícil, pelo fato de existirem mais lixeiras, portanto também mais recursos. Porém, não foi criada uma classe para cada nível, e sim uma restrição para que se o nível for fácil, os botões dos lixos Orgânico e Não Reciclável ficam ocultos, e se o nível for médio, apenas o botão Não Reciclável fica oculto. Na seguinte linha de código contido nas páginas 41 e 43 do documento Código fonte dá pra ver como isso foi feito:

```
if(nivel < 1)
{
btnOrganico.setVisible(false);
lblT.setVisible(false);
}

if(nivel < 2)
{
btnNaoReciclavel.setVisible(false);
lblY.setVisible(false);
}
```

Visto que nível = 0 se refere ao nível fácil, nível = 1 se refere ao nível médio e nível = 2 se refere ao nível difícil.

Abaixo segue então o print-screen da tela do jogo, sem nenhuma alteração ainda feita.



Sem mais, vamos clicar no botão iniciar para ver o que acontece.

Percebe-se que os botões “Iniciar” e “Voltar” se ocultaram. Além disso, todos os botões de lixeiras foram habilitados e agora, cada clique que o usuário lançar sobre eles afetará diretamente no resultado do jogo.

Nota-se também que a barra de progresso começa a se decrementar, e como foi bem especificado nas dicas do jogo, se ela chegar ao fim o jogo termina com derrota para o jogador! Um segredo muito interessante que está escondido do usuário, é que a cada acerto a barra de progresso vai ficando mais rápida. Isso é pouco perceptível enquanto joga, mas no final acaba fazendo uma diferença muito grande.

Ao lado da barra de progresso se abriram as informações que vão se alterando durante o jogo, são elas “Pontos: 0” que vai aumentando ou diminuindo conforme a variável que controla os pontos vai se alterando, “Palavras: 0” que acrescenta 1 a cada palavras respondida, “Acertos: 0” que acrescenta 1 a cada acerto e “Erros: 0” que acrescenta 1 a cada erro. Essas informações são importantes pois pode ajudar o usuário a saber como que ele está indo no jogo, mas é bom tomar cuidado pois enquanto se perde tempo olhando as informações, a barra de progresso não para de cair.

E pra terminar a lista de novidades ao clicar no botão “Iniciar”, temos a palavra e a imagem que apareceu logo acima dos lixos. Essas palavras e imagens serão alteradas conforme o usuário vai respondendo cada lixo. É muito importante notar

que cada imagem foi cuidadosamente recortada, removido o fundo e salvo em formato .PNG. Os lixos não se repetem e isso foi feito com a utilização de listas em JAVA. Conforme a palavra (que corresponde a uma posição de um vetor) vai sendo sorteada, esta é adicionada na lista e na próxima palavra é feita uma verificação para ver se a nova palavra sorteada não contém na lista. Caso não, a palavra é mostrada para o usuário normalmente, mas caso sim ela não é mostrada e o programa sorteia novamente, até encontrar uma posição no vetor que não contenha na lista. Esta parte do código está no método sorteio(), localizado nas páginas 29, 30 e 31 do arquivo Código Fonte.pdf.

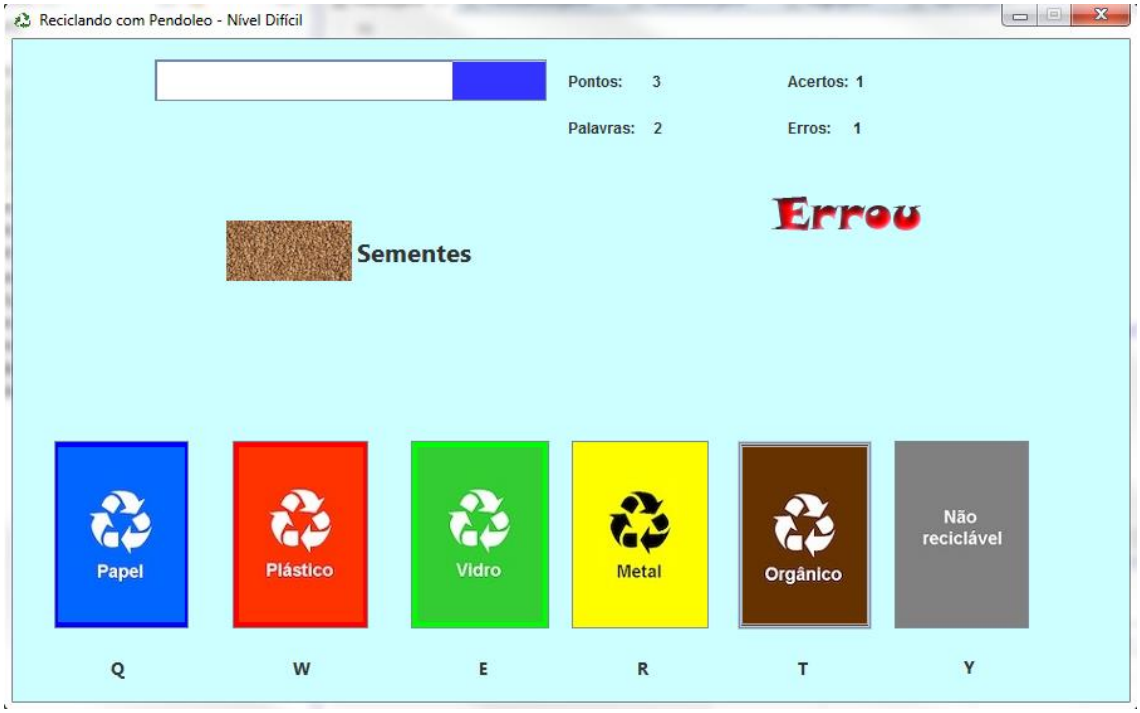
Não temos muito tempo para pensar pois a barra de progresso continua descendo, então temos que verificar qual é a lixeira correspondente a “Folhas”, que no caso é o lixo Orgânico. Vamos responder e mostrar aqui o que acontece.

Conforme citado nas dicas, ganhei 5 pontos, ganhei mais um pouco de tempo para pensar na próxima palavra e logo em seguida já apareceu outro lixo para responder.

Perceba na tela que meus pontos foram alterados, minha quantidade de acertos também e as palavras respondidas.

Além disso, uma mensagem para informar se o usuário acertou ou errou também aparece, nesse caso eu acertei.

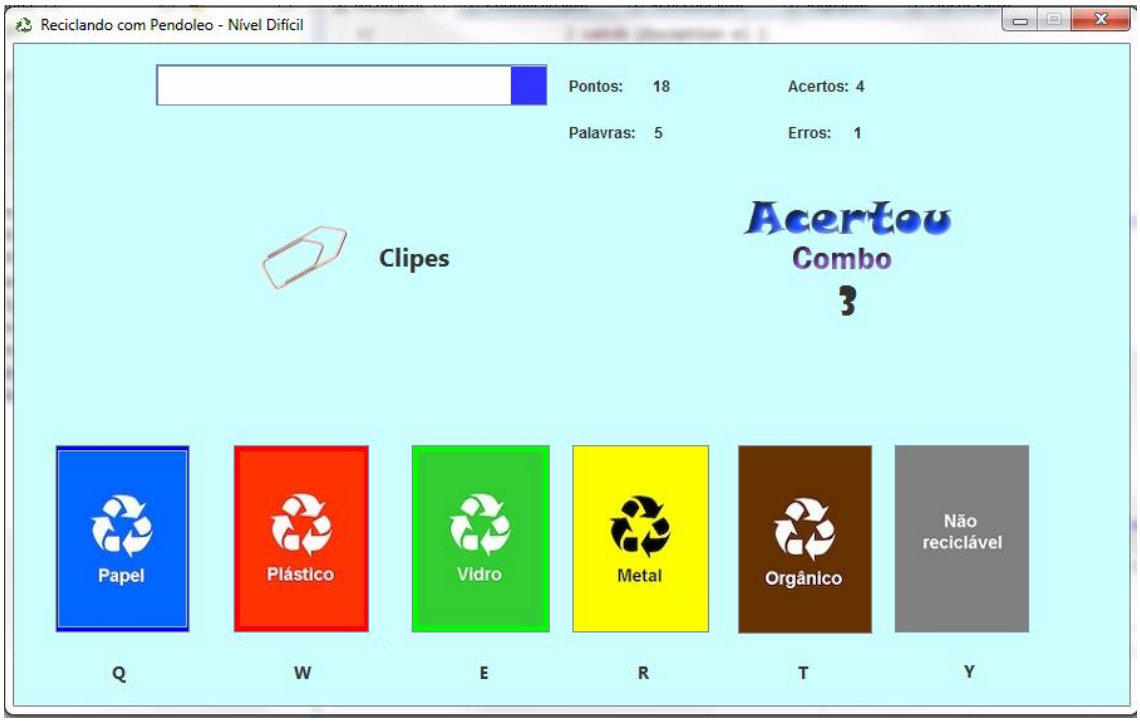
Agora tenho a palavra “Lata de verniz”, que se trata de um lixo não reciclável. Porém vou responder que é um vidro só para ver como que o programa vai responder ao meu erro. Veja no print-screen que segue no começo da próxima página.



Novamente como orientado nas dicas do jogo, perdi um pouco de tempo da minha barra de progresso. Além desse malefício, também perdi 2 pontos. Uma mensagem informando o erro também foi mostrada, além da alteração dos números de palavras respondidas e de erros.

Pensa que acabou o tanto de funcionalidades do jogo? Ainda não. Vamos mostrar agora o contador de combo, que ajuda tanto aquelas pessoas que dão sorte de pegar uma sequencia de palavras fáceis. Ele funciona da seguinte maneira: a partir de 3 palavras acertadas de forma seguida, ele altera o valor de pontos ganhos em um acerto, ou seja, cada lixo correto você passa a ganhar 7 pontos. Porém não pense que isso dura até o fim da partida, obviamente se tratando de um combo, ao

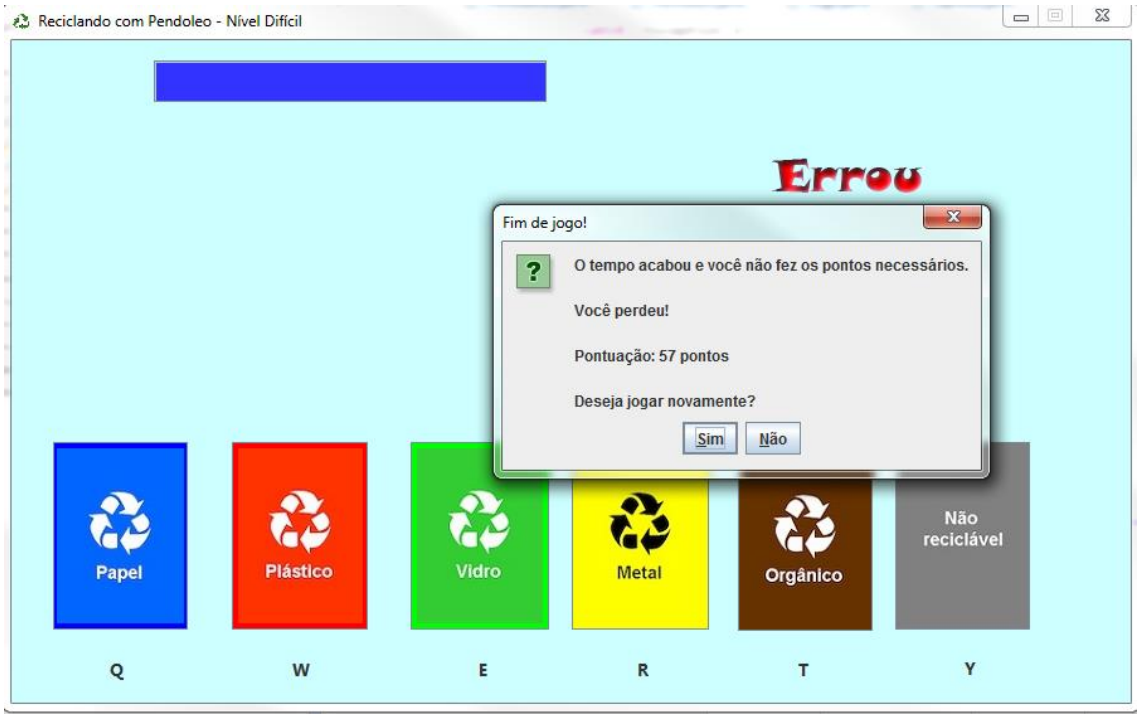
errar você volta a ganhar 5 pontos cada acerto. Enquanto você estiver dentro de um combo, o jogo avisará com a mensagem que é mostrada no print da próxima página.



Vimos na tela acima que além da mensagem mostrada, ele informa a quantidade de palavras que você já acertou de forma seguida. É bom tentar aproveitar esse recurso do jogo, ainda mais no nível difícil, mas para isso deve ter sorte e competência para acertar uma boa quantidade de palavras seguidas. Você pode acertar de 3 a 90 palavras seguidas, que é o número máximo de palavras do jogo. Veja um exemplo no qual o usuário está utilizando muito bem o recurso combo.

Agora vamos começar a ver uma parte do jogo que foi tratado com muitos detalhes: o fim. Nesse jogo, se ganha é compensado com um frame animado te parabenizando da conquista. Se perde, o programa informa através de uma caixa de mensagem o motivo da sua derrota, se foi porque a barra de progresso acabou, se foi porque você não conseguiu os pontos necessários, ou então se foi pelos dois motivos. Vamos fazer os testes.

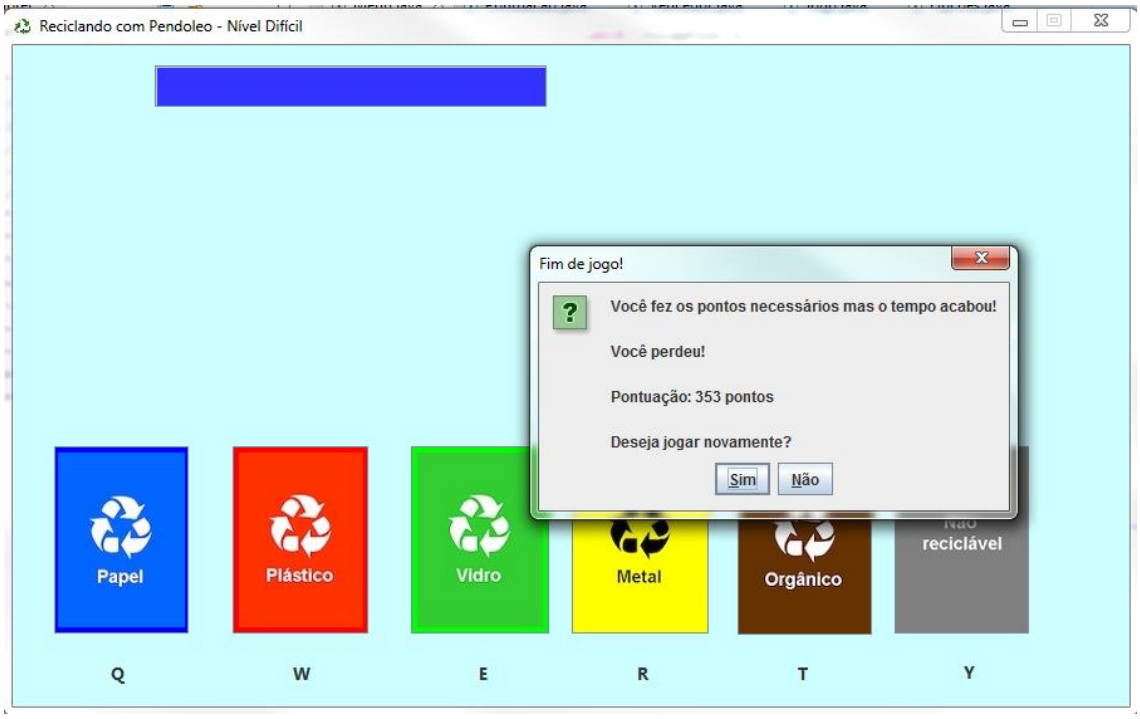
Primeiro vou simular a situação em que a barra de progresso chega ao fim e eu também não consegui somar os pontos necessários. Abaixo segue a tela da mensagem que o programa informa.



Vejamos que ele informa corretamente o ocorrido e nos oferece a jogar novamente. Caso você escolha “Sim” não se preocupe que o jogo ainda oferece um tempo para que você descanse até você decidir clicar no botão “Iniciar” novamente. Se você escolher “Não”, será chamada novamente a classe Menu.java exibindo o menu principal do game.

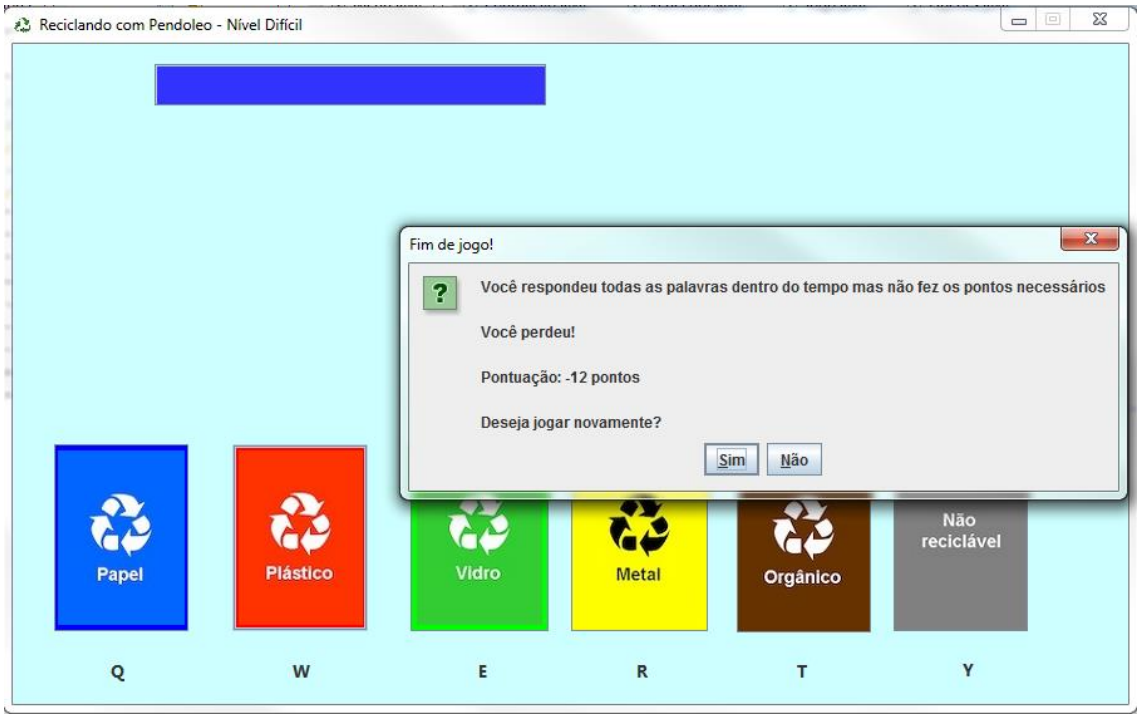
Então vamos continuar a simulação de todas as formas que existe para sair derrotado desse jogo. Se você chegou aos pontos necessários de acordo com o nível que escolheu e vai tentar ser esperto, apenas esperando o tempo acabar para você não correr o risco de errar alguma palavra e perder pontos, vamos ver no que isso vai dar.

Provavelmente quem fez isso não chegou a ler as regras antes do começo do jogo, que fala claramente que o tempo não pode chegar ao fim que o jogo declarará derrota para o jogador. Vamos ver qual é a mensagem que o jogo mostra quando isso acontece.



Conforme dito, ele mostra que você fez os pontos necessário mas o tempo chegou ao fim, portanto você perdeu. Essa situação é bem comum de acontecer, pois fazer os pontos necessários não é tão difícil ainda mais com a ajuda do combo, mas o problema é que a cada erro o usuário perde muito da barra de progresso e além de tudo ela vai ficando mais rápida a cada acerto.

Além dessas duas possibilidades de derrota citadas acima, tem ainda a mais incomum entre elas. Esta é aquela que você conseguiu driblar o tempo, respondeu todas as palavras antes que o tempo acabasse, mas mesmo assim não conseguiu fazer os pontos. Normalmente, quando isso acontece é porque a pessoa não fez proveito do combo e também quis responder as palavras de forma muito rápida sem ter certeza muitas vezes. Veja na próxima página a tela de quando esse tipo de derrota acontece.



Apesar de muito raro esse tipo de situação acontecer, não é impossível. Normalmente, se você consegue responder a todas as palavras você consegue também os pontos necessários, principalmente no nível difícil.

Agora por último temos talvez a parte mais esperada do jogo: o fim, porém com vitória para o jogador. Ao verificar que o jogador somou os pontos necessários e terminou de responder a todas as palavras, o jogo automaticamente se encerra e chama a classe Vencedor.java. Essa classe não contém muitas informações, apenas a pontuação do jogador e uma imagem o parabenizando pela conquista. Além logicamente da opção do jogador jogar novamente ou sair e voltar para o menu. Seguindo os cálculos do jogo, a pontuação máxima que pode ser alcançada é de 626 pontos, visto que se o jogador acertar todas as palavras ele ganhará 10 pontos nas duas primeiras palavras e depois 7 em cada até a palavra 90, que resultará na conta (88 * 7 + 10) = 626. Porém por se tratar de um jogo que exige um raciocínio muito rápido, é muito difícil atingir essa marca e fica como um desafio a mais para quem já conseguiu superar todos os níveis do jogo.

A seguir temos o print da tela após conseguir a vitória.

Gostou?!

O jogo está disponibilizado para download no site Media Fire no formato executável .jar, através do seguinte link:

http://download1591.mediafire.com/0jxqwipl6lwg/ifx0h5q932cppfi/ReciclandoComPenDoLeo.jar

Basta acessar o link acima e automaticamente o arquivo será descarregado.

Além desses dois links, também disponibilizamos o download dos arquivos do jogo, contendo todo o projeto para fins de informação e conhecimento também. Esse link também está no site Media Fire no link a seguir:

http://download1491.mediafire.com/6nu88rtsi33g/u2u3433abxc6sbv/APS.zip

10. Bibliografia

BIBLIOGRAFIA

Sindico Net: http://www.sindiconet.com.br/6857/Informese/Coleta-Seletiva/Lista-de-materiais-reciclaveis-e-naoreciclaveis

Ecycle: http://www.ecycle.com.br/component/content/article/44-guia-da-reciclagem/706-conheca-os-tipos-de-plastico.html

Sua Pesquisa: http://www.suapesquisa.com/reciclagem/reciclagem_de_vidro.htm

Gesto: http://gesto.prdf.mpf.mp.br/legislacao-e-outros-documentos/o_que_e_lixo_organico_e_inorganico.pdf

Revista Escola: http://revistaescola.abril.com.br/fundamental-1/qual-diferenca-lixo-organico-inorganico-732750.shtml

Se liga no Lixo: https://seliganolixo.wordpress.com/por-que-o-lixo-e-um-problema/tipos-de-lixo/

Java2s: http://www.java2s.com/Tutorial/Java/0160__Thread/Addadelay.htm

Caelum: http://www.caelum.com.br/apostila-java-estrutura-dados/listas-ligadas/#5-4-testes

Oracle: https://docs.oracle.com/javase/tutorial/essential/concurrency/runthread.html

.

11. Ficha de Atividades Práticas Supervisionadas

UNIP UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME: LEONARDO PEREIRA MOREIRA TURMA: CC4Q41 RA: A96730-3

CURSO: 13701 - CIÊNCIA DA COMPUTAÇÃO CAMPUS: RNG - SANTOS - RANGEL TURNO: NOITE

CÓDIGO DA ATIVIDADE: 385X - ATIVIDADES PRÁTICAS SUPERVISIONADAS SEMESTRE: 4º ANO GRADE: 2014/2

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
10/09/2014	Reunião para discutir o tema e o software usado para fazer o jogo	3 horas	Leonardo Pereira Moreira		
10/09/2014	Reunião para definir a forma como o jogo será desenvolvido.	2 horas			
10/09/2014	Divisão de tarefas entre os participantes do grupo	1 hora			
12/09/2014	Começo do desenvolvimento da classe Jogo.java	3 horas			
15/09/2014	Desenvolvimento do jogo, definindo os padrões de cores e botões	3 horas			
15/09/2014	Desenvolvimento da classe menu.java	6 horas			
16/09/2014	Começando o desenvolvimento da classe Opcoes.java	2 horas			
25/09/2014	Reunião para definir as regras do jogo	3 horas			
01/10/2014	Desenvolvimento da classe Opcoes.java	3 horas			
05/10/2014	Desenvolvimento da classe Pontuacao.java	3 horas			
20/10/2014	Desenvolvimento da classe Vencedor.java	3 horas			
22/10/2014	Desenvolvimento da aplicação, usando o eclipse 4.3	3 horas			
10/11/2014	Desenvolvimento da classe Jogo.java	3 horas			
17/11/2014	Reunião para concluir o jogo e fazer a divisão das atividades do arquivo .pdf	2 horas			
23/11/2014	Trabalhando em cima da parte teórica da aplicação.	8 horas			
25/11/2014	Conclusão da atividade	2 horas			

(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.

TOTAL DE HORAS ATRIBUÍDAS: 50 Horas

AVALIAÇÃO: _____
Aprovado ou Reprovado

NOTA: _____

DATA: ____/____/_____

_____
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

UNIP
UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME: VICTOR BRUNO DOS SANTOS PAIVA **TURMA:** SI3P41 **RA:** T13588-5

CURSO: 15901-SISTEMAS DE INFORMACAO **CAMPUS:** RNG - SANTOS - RANGEL **TURNO:** NOITE

CÓDIGO DA ATIVIDADE: 378X - ATIVIDADES PRÁTICAS SUPERVISIONADAS **SEMESTRE:** 3º **ANO GRADE:** 2014/2

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
10/09/2014	Reunião para discutir o tema e o software usado para fazer o jogo	3 horas	Victor Bruno Dos Santos Paiva		
10/09/2014	Reunião para definir a forma como o jogo será desenvolvido.	2 horas			
10/09/2014	Divisão de tarefas entre os participantes do grupo	1 hora			
12/09/2014	Começo do desenvolvimento da classe Jogo.java	3 horas			
15/09/2014	Desenvolvimento do jogo, definindo os padrões de cores e botões	3 horas			
15/09/2014	Desenvolvimento da classe menu.java	6 horas			
16/09/2014	Começando o desenvolvimento da classe Opcoes.java	2 horas			
25/09/2014	Reunião para definir as regras do jogo	3 horas			
01/10/2014	Desenvolvimento da classe Opcoes.java	3 horas			
05/10/2014	Desenvolvimento da classe Pontuacao.java	3 horas			
20/10/2014	Desenvolvimento da classe Vencedor.java	3 horas			
22/10/2014	Desenvolvimento da aplicação, usando o eclipse 4.3	3 horas			
10/11/2014	Desenvolvimento da classe Jogo.java	3 horas			
17/11/2014	Reunião para concluir o jogo e fazer a divisão das atividades do arquivo .pdf	2 horas			
23/11/2014	Trabalhando em cima da parte teórica da aplicação.	8 horas			
25/11/2014	Conclusão da atividade	2 horas			

(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.

TOTAL DE HORAS ATRIBUÍDAS: 50 Horas

AVALIAÇÃO: _____
Aprovado ou Reprovado

NOTA: _____

DATA: ____/____/_____

_____
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

UNIP
UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME: MATHEUS FELIPE DOS PASSOS E PAZ **TURMA:** CC4Q41 **RA:** B57IAJ-0

CURSO: 13701 - CIÊNCIA DA COMPUTAÇÃO **CAMPUS:** RNG - SANTOS - RANGEL **TURNO:** NOITE

CÓDIGO DA ATIVIDADE: 385X - ATIVIDADES PRÁTICAS SUPERVISIONADAS **SEMESTRE:** 4º **ANO GRADE:** 2014/2

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
10/09/2014	Reunião para discutir o tema e o software usado para fazer o jogo	3 horas	Matheus Felipe dos Passos e Paz		
10/09/2014	Reunião para definir a forma como o jogo será desenvolvido.	2 horas			
10/09/2014	Divisão de tarefas entre os participantes do grupo	1 hora			
12/09/2014	Começo do desenvolvimento da classe Jogo.java	3 horas			
15/09/2014	Desenvolvimento do jogo, definindo os padrões de cores e botões	3 horas			
15/09/2014	Desenvolvimento da classe menu.java	6 horas			
16/09/2014	Começando o desenvolvimento da classe Opcoes.java	2 horas			
25/09/2014	Reunião para definir as regras do jogo	3 horas			
01/10/2014	Desenvolvimento da classe Opcoes.java	3 horas			
05/10/2014	Desenvolvimento da classe Pontuacao.java	3 horas			
20/10/2014	Desenvolvimento da classe Vencedor.java	3 horas			
22/10/2014	Desenvolvimento da aplicação, usando o eclipse 4.3	3 horas			
10/11/2014	Desenvolvimento da classe Jogo.java	3 horas			
17/11/2014	Reunião para concluir o jogo e fazer a divisão das atividades do arquivo .pdf	2 horas			
23/11/2014	Trabalhando em cima da parte teórica da aplicação.	8 horas			
25/11/2014	Conclusão da atividade	2 horas			

(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.

TOTAL DE HORAS ATRIBUÍDAS: 50 Horas

AVALIAÇÃO: _____
Aprovado ou Reprovado

NOTA: _____

DATA: ____/____/_____

_____
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

UNIP
UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME: MATHEUS RODRIGUES MARTINS PEREIRA **TURMA:** CC4Q41 **RA:** B73ABD-5

CURSO: 13701 - CIÊNCIA DA COMPUTAÇÃO **CAMPUS:** RNG - SANTOS - RANGEL **TURNO:** NOITE

CÓDIGO DA ATIVIDADE: 385X - ATIVIDADES PRÁTICAS SUPERVISIONADAS **SEMESTRE:** 4º **ANO GRADE:** 2014/2

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
10/09/2014	Reunião para discutir o tema e o software usado para fazer o jogo	3 horas	Matheus Rodrigues Martins Pereira		
10/09/2014	Reunião para definir a forma como o jogo será desenvolvido.	2 horas			
10/09/2014	Divisão de tarefas entre os participantes do grupo	1 hora			
12/09/2014	Começo do desenvolvimento da classe Jogo.java	3 horas			
15/09/2014	Desenvolvimento do jogo, definindo os padrões de cores e botões	3 horas			
15/09/2014	Desenvolvimento da classe menu.java	6 horas			
16/09/2014	Começando o desenvolvimento da classe Opcoes.java	2 horas			
25/09/2014	Reunião para definir as regras do jogo	3 horas			
01/10/2014	Desenvolvimento da classe Opcoes.java	3 horas			
05/10/2014	Desenvolvimento da classe Pontuacao.java	3 horas			
20/10/2014	Desenvolvimento da classe Vencedor.java	3 horas			
22/10/2014	Desenvolvimento da aplicação, usando o eclipse 4.3	3 horas			
10/11/2014	Desenvolvimento da classe Jogo.java	3 horas			
17/11/2014	Reunião para concluir o jogo e fazer a divisão das atividades do arquivo .pdf	2 horas			
23/11/2014	Trabalhando em cima da parte teórica da aplicação.	8 horas			
25/11/2014	Conclusão da atividade	2 horas			

(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.

TOTAL DE HORAS ATRIBUÍDAS: 50 Horas

AVALIAÇÃO: _____
Aprovado ou Reprovado

NOTA: _____

DATA: ____/____/_____

_____
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO