LUCAS HENRIQUE RONQUI DE SOUZA

Desenvolvimento de Aplicativo Educativo para Plataforma Android

Assis

2012

LUCAS HENRIQUE RONQUI DE SOUZA

Desenvolvimento de Aplicativo Educativo para Plataforma Android

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação.

ORIENTADOR: Felipe Alexandre Cardoso Pazinatto

ÁREA DE CONCENTRAÇÃO: Informática

Assis

2012

FICHA CATALOGRÁFICA

SOUZA, Lucas Henrique Ronqui

Desenvolvimento de Aplicativo Educativo para Plataforma Android / Lucas Henrique Ronqui de Souza. Fundação Educacional do Município de Assis – FEMA – Assis, 2012.

113 p.

Orientador: Felipe Alexandre Cardoso Pazinatto

Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de Assis – IMESA.

1. Android. 2. Dengue. 3. Jogo.

CCD: 001.6

Biblioteca da FEMA

DESENVOLVIMENTO DE APLICATIVO EDUCATIVO PARA PLATAFORMA ANDROID

LUCAS HENRIQUE RONQUI DE SOUZA

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação, analisado pela seguinte comissão examinadora:

Orientador: Felipe Alexandre Cardoso Pazinatto

Analisador (1): Luiz Ricardo Begosso

Assis

2012

DEDICATÓRIA

Dedico este trabalho a todos que me ajudaram a conclui-lo, familiares, professores, colegas de serviço e especialmente minha noiva.

AGRADECIMENTOS

Em primeiro lugar agradeço a Deus por ter me dado sabedoria, saúde e paciência para realizar este trabalho e outros projetos que almejo em minha vida.

Ao meu orientador Felipe Alexandre Cardoso Pazinatto, pelas aulas e teorias dadas nos vários encontros de sábado que me ajudaram a concluir este desafio pessoal.

A minha família por sempre estarem presentes quando era necessário.

Agradeço também aos meus colegas de serviço por me ajudarem com as pesquisas e apoiarem minha iniciativa.

E por último, agradeço também a minha futura esposa por sempre estar ao meu lado nos momentos tristes e felizes além de ter me ajudado na correção e revisão do trabalho.

RESUMO

Devido ao grande avanço da tecnologia, os dispositivos móveis de hoje em dia possuem muitos recursos a serem explorados pelos programadores. Com a chegada do novo S.O da Google, o Android, ficou ainda mais fácil criar seu próprio programa para celular, tablet, etc.

O Android tornou-se rapidamente popular graças à sua capacidade de explorar os recursos disponíveis nos celulares atualmente e por causa de seu custo. Mas que custo? O SDK do Android não é gratuito!? Por isso mesmo, por ser gratuito ele cresceu numa velocidade impressionante.

A proposta deste trabalho será o desenvolvimento de um aplicativo para a plataforma Android abordando um tema que tem causado muita dor de cabeça para a população brasileira, e de outros países também, a dengue. O aplicativo será um jogo que tem por objetivo dar dicas sobre o combate a dengue.

Palavras-chave: Android, Dengue, Jogo.

ABSTRACT

Due to the great advancement of technology, the mobile devices nowadays have many resources to be exploited by the programmers. With the arrival of the new OS of Google, the Android, it's even easier to create your own program for mobile, tablet, etc.

The Android quickly became popular thanks to the ability to exploit resources available on mobile phones nowadays and because of its cost. But what cost? Isn’t the Android SDK free? Therefore, for being free it grew up with incredible speed.

So the purpose of this paper is to develop an application for the Android platform broaching a theme that has caused much headache for the Brazilian population, and other countries as well, the dengue fever. The app will be a game that aims to give tips on combating the dengue fever.

Palavras-chave: Android, Dengue Fever, Game.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1. INTRODUÇÃO

Já faz um tempo que o número de celulares e dispositivos móveis ultrapassou a quantidade de habitantes no Brasil. Atualmente, segundo a ANATEL, nosso país possui cerca de 195.821.584 habitantes contra 247.618.048 de celulares.

Com o avanço tecnológico, os celulares não estão apenas restritos a realizar e receber chamadas. O fato é que essa é uma realidade que já está ultrapassada. Hoje em dia os celulares são mini-computadores com muito mais recursos que os antigos e bem menores.

Mas não é só no hardware que os dispositivos móveis evoluíram, os softwares também sofreram um grande avanço. Isso ocorreu devido às novas possibilidades trazidas pelos novos hardwares e também pelo novo Sistema Operacional desenvolvido pela Google, o Android.

Mas por que o Android é responsável por esse avanço? É simples, por ele ser um sistema aberto, é possível instalar softwares de outros desenvolvedores e não apenas daquele que é responsável pelo S.O.

Existe uma grande variedade de aplicativos disponíveis para Android. Estes podem ser baixados pelo Google Play (antigo Android Market), onde estão divididos em 26 categorias que abordam desde a área de entretenimento até a saúde. Esta última, apesar de sua importância, nem sempre faz muito sucesso com os usuários de Android.

Existem vários aplicativos que explicam e dão dicas sobre saúde, nutrição, condicionamento físico, etc. Infelizmente poucos desses aplicativos são voltados à saúde pública, ou se existem são conhecidos por uma parcela muito pequena da população (de 5 a 20 pessoas possuem o aplicativo). Isso é alarmante, pois tais programas foram desenvolvidos para ajudar a tratar ou combater várias doenças que são transmitidas facilmente e que muitas vezes, por falta de conhecimento, chegam a causar até óbito.

1.1 OBJETIVO

Este trabalho tem por objetivo a elaboração de um jogo educativo para a plataforma Android, que aborde uma das principais doenças que atualmente afetam o Brasil, a dengue. O jogo deve ensinar como combater os mosquitos transmissores, o *Aedes Aegypit* e o *Aedes Albopictus*, e também quais os procedimentos que devem ser tomados caso haja alguma suspeita da doença.

1.2 PÚBLICO ALVO

O público alvo do trabalho é toda a população brasileira que utiliza algum dispositivo móvel com Android, pois quanto maior o número de pessoas conscientizadas sobre os meios de prevenção, da qual o principal é a destruição de criadouros, melhor será o resultado dessa "luta" contra a dengue.

1.3 JUSTIFICATIVA

Atualmente o combate a dengue mostra-se bastante eficiente, entretanto a necessidade de se tomar uma medida alternativa para melhorar a conscientização da população brasileira é bem visível. O jogo deve abordar uma das mais eficientes formas de controlar o contágio da doença, a destruição dos possíveis criadouros que em muitos casos estão presentes nos quintais das casas.

2. DENGUE

2.1 HISTÓRIA

Segundo Halstead, o vírus causador da dengue em humanos evoluiu como um parasita dos primatas [1].

Os primeiros registros de dengue no mundo foram feitos no fim do século 18, na ilha de Java, no Sudoeste Asiático, e na Filadélfia, Estados Unidos, mas somente no século passado (século 20), a dengue foi reconhecida como doença pela Organização Mundial da Saúde [2].

Já no Brasil, a disseminação da dengue iniciou-se na época da colonização com a vinda dos navios que traziam escravos da áfrica. O mosquito deve ter implantado seus ovos nos reservatórios de água sobrevivendo todo o caminho até aqui. Benseñor[3] diz que o primeiro caso de dengue no Brasil surgiu em 1865, na cidade de Recife. Sete anos depois, em Salvador uma epidemia de dengue levou a 2.000 mortes.

2.2 O QUE É A DENGUE?

A dengue é uma doença causada por quatro sorotipos do vírus dela. Eles foram nomeados como Dengue do tipo 1, 2, 3 e 4. Essa variação não foi descoberta logo no inicio. Os sorotipos 1 e 2 foram descobertos em 1960, o sorotipo 3 foi descoberto em 1961 e, por fim, o tipo 4 em 1963 [4].

A única maneira de descobrir qual sorotipo uma pessoa “carrega” é através da análise sanguínea. Por serem semelhantes, os sintomas e a evolução da doença seguem o mesmo padrão, sofrendo algumas mudanças dependendo do organismo da pessoa. Muitas vezes a dengue é confundida com uma gripe forte.

Além disso, a dengue só pode ser adquirida através da picada do mosquito transmissor, o *Aedes aegypti* ou o *Aedes albopictus*, contaminado com o vírus. Dentre os dois o mais conhecido é o *Aedes aegypti* que é um pouco menor que o pernilongo comum e possui listras brancas no corpo.

2.3 SINTOMAS

Como dito anteriormente, os sintomas podem variar de pessoa a pessoa, mas os principais sintomas da dengue comum são:

- Febre alta com início súbito.
- Mialgia (dor no muscular).
- Artralgia (dor nas articulações).
- Dor Retro Ocular.
- Manchas Avermelhadas na Pele.
- Cefaléia (dor de cabeça).
- Náuseas e vômitos.
- Perda do paladar e apetite.
- Tonturas.
- Extremo cansaço.
- Moleza e dor no corpo.

Já a dengue hemorrágica os sintomas são os mesmos da dengue comum. A diferença ocorre quando acaba a febre e começam a surgir os sinais de alerta:

- Dores abdominais fortes e contínuas.
- Vômitos persistentes.
- Pele pálida, fria e úmida.
- Sangramento pelo nariz, boca e gengivas.
- Manchas vermelhas na pele.

- Sonolência, agitação e confusão mental.
- Sede excessiva e boca seca.
- Pulso rápido e fraco.
- Dificuldade respiratória.
- Perda de consciência

Conforme informações disponíveis no Site da Dengue, mantido pela AJA Brasil (Associação Jovem Aprendiz), na dengue hemorrágica o quadro clínico se agrava rapidamente, apresentando sinais de insuficiência circulatória e choque, podendo levar a pessoa à morte em até 24 horas [2]. De acordo com estatísticas do Ministério da Saúde, cerca de 5% das pessoas com dengue hemorrágica morrem. A Figura 1 abaixo ilustrará melhor a diferença entre os sintomas da dengue comum e a hemorrágica.

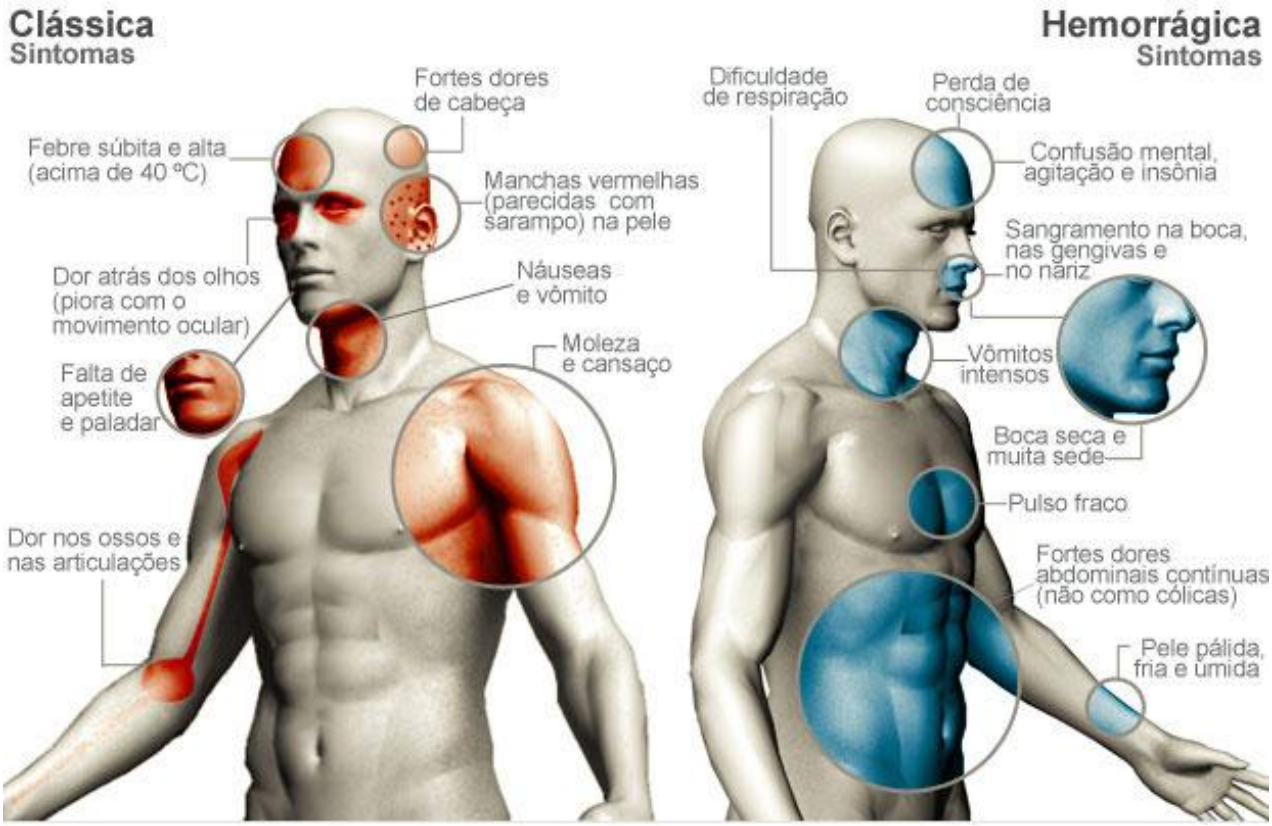

Figura 1: Dengue Comum x Hemorrágica. Fonte [2].

2.4 COMO COMBATER?

Existem vários meios de combater a dengue, dentre elas o principal é a eliminação dos ovos do *Aedes aegypti* e de seus possíveis criadouros que vão desde caixas d'água a tampinhas de garrafa. Ou seja, os criadouros são todos aqueles recipientes que possam acumular água, não importando tamanho ou forma. A Figura 2 mostra as atitudes básicas que se deve tomar para a eliminação de alguns criadouros presentes em residências.

Figura 2: Dicas para combater o mosquito e os focos de larvas. Fonte [2].

Atualmente, para que haja esse tipo de controle primeiramente é feito um serviço de conscientização da população, através de panfletagem, palestras, atividades em escolas e nos principais veículos de comunicação como televisão, rádio, internet, etc. Além disso, complementando o trabalho de conscientização, existem os Agentes de Combate fazendo visitas às casas auxiliando a população a fazer o controle e explicando questões relacionadas à dengue e outras doenças.

2.5 TRATAMENTO

Em uma cartilha, o Ministério da Saúde informa:

> Normalmente a doença dura de 5 a 7 dias. Quem está com dengue deve ficar em repouso e beber muita água. Não há um tratamento específico para a doença. As medicações utilizadas são analgésicos (remédios para aliviar a dor) e antitérmicos (para diminuir a febre). No entanto, nunca se deve tomar medicamentos sem orientação médica.[5].

3. ANDROID

Como dito anteriormente, o mercado de celulares e dispositivos móveis encontra-se em constante crescimento. Segundo WROBLEWSKI [6] nascem cerca de 370 mil crianças por dia no mundo, enquanto são ativados em torno de 562 mil dispositivos móveis que utilizam o iOS, maior rival do Android que o superou chegando a marca de 700 mil dispositivos móveis ativados por dia, observe a Figura 3.

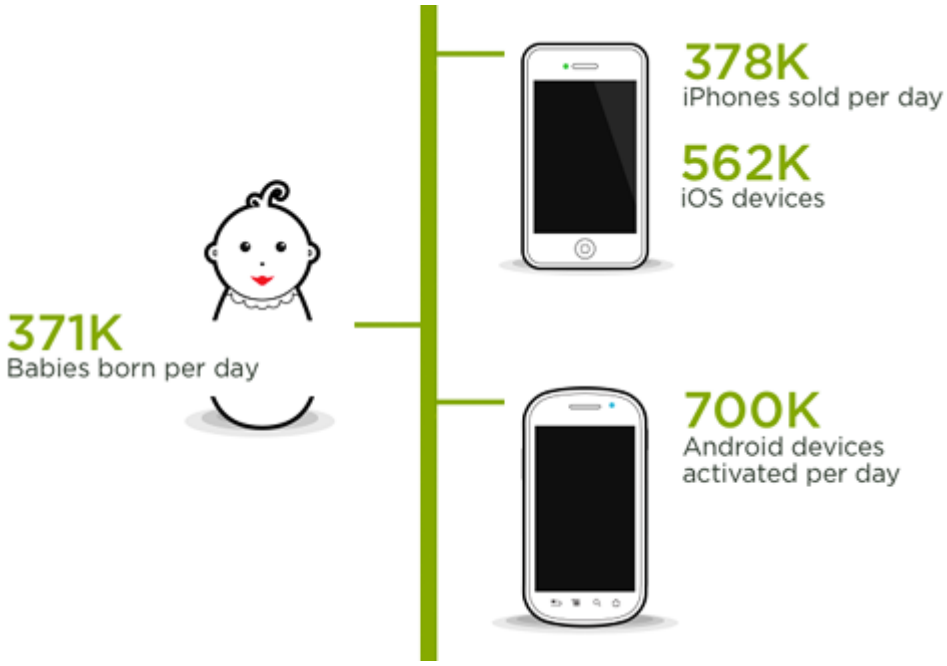

Figura 3 – Bebês x Mobiles. Fonte [6].

Além disso, segundo uma pesquisa feita pela Nielsen [7] o Android venceu seu concorrente durante três meses consecutivos. Observe a Figura 4, mostrando seu incrível crescimento diante desse gigante do mercado tecnológico, a *Apple*.

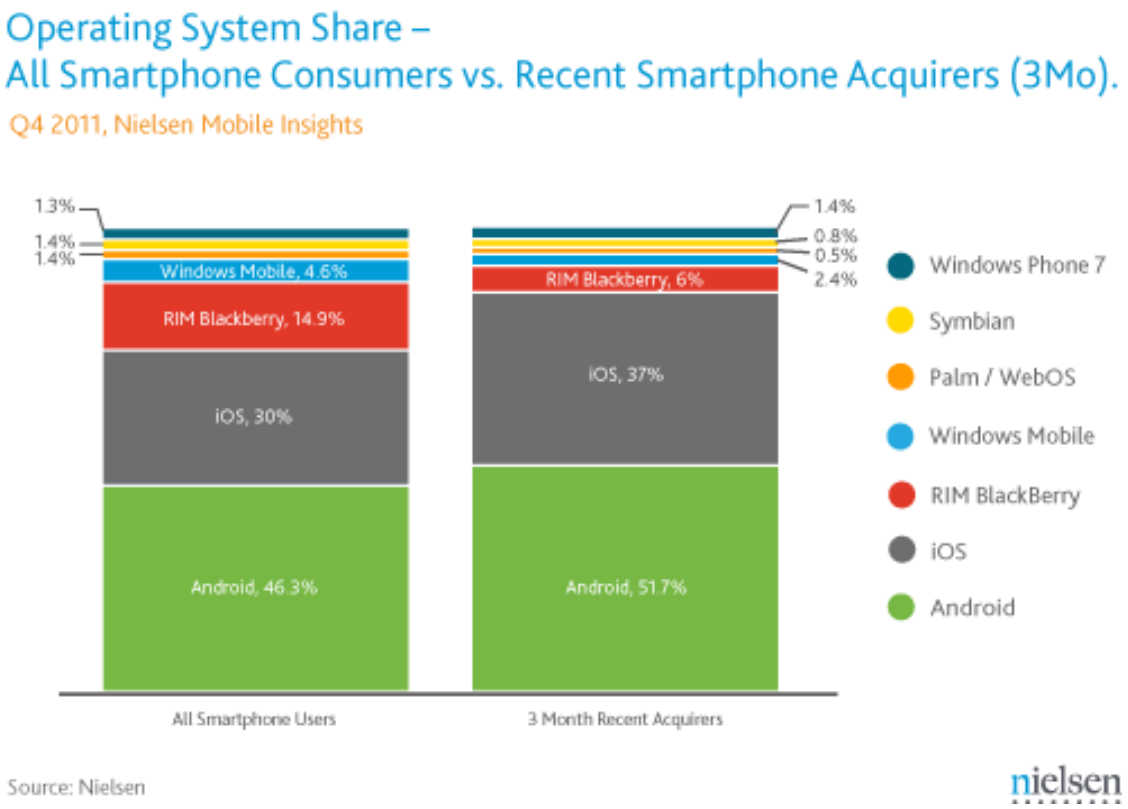

Figura 4 – Todos Consumidores de Smartphones vs. Recentes Compradores de Smartphones. [7].

Conclui-se então que a distância entre *Apple* e Android está cada vez menor, apesar da Apple ainda estar na liderança, porém por ser *Open Source* muitos programadores preferem desenvolver para plataforma Android.

3.1 HISTÓRIA

O Android é um produto criado em conjunto com a Open Handset Alliance (OHA) liderado pela Google [8]. A OHA é formada por operadoras de celular, fabricantes de aparelho, empresas de semicondutores, empresas de software e empresas de comercialização. Ou seja, por todas as

estruturas envolvidas no processo de telefonia móvel [9]. A Figura 3 apresenta o logotipo de algumas das empresas que formam a OHA.

Figura 5: Membros OHA. Fonte: [10].

Segundo PEREIRA [11]:

> O Android é uma plataforma para tecnologia móvel completa, envolvendo um pacote com programas para celulares, já com um sistema operacional, middleware, aplicativos e interface do usuário.

> Android foi construído com a intenção de permitir aos desenvolvedores criar aplicações móveis que possam tirar total proveito do que um aparelho portátil pode oferecer. Foi construído para ser verdadeiramente aberto. Por exemplo, uma aplicação pode apelar a qualquer uma das funcionalidades de núcleo do telefone, tais como efetuar chamadas, enviar mensagens de texto ou utilizar a câmera, que permite aos desenvolvedores adaptarem e evoluírem cada vez mais estas funcionalidades.

Por ser aberto, é possível utilizar softwares feitos por qualquer desenvolvedor, sem depender da empresa fabricante, isso fez com que o Android se tornasse cada vez mais popular.

Mas como fazer um programa para utilizar no dispositivo móvel com Android? Existem várias maneiras de se desenvolver, entretanto, para quem ainda não está familiarizado com o ambiente Android, é indicado iniciar-se utilizando o IDE Eclipse. Para isto você deve instalar um *plugin* disponível no site oficial do Android.

3.2 CAMADAS

A seguir, a Figura 6 mostra as camadas presentes na plataforma Android.

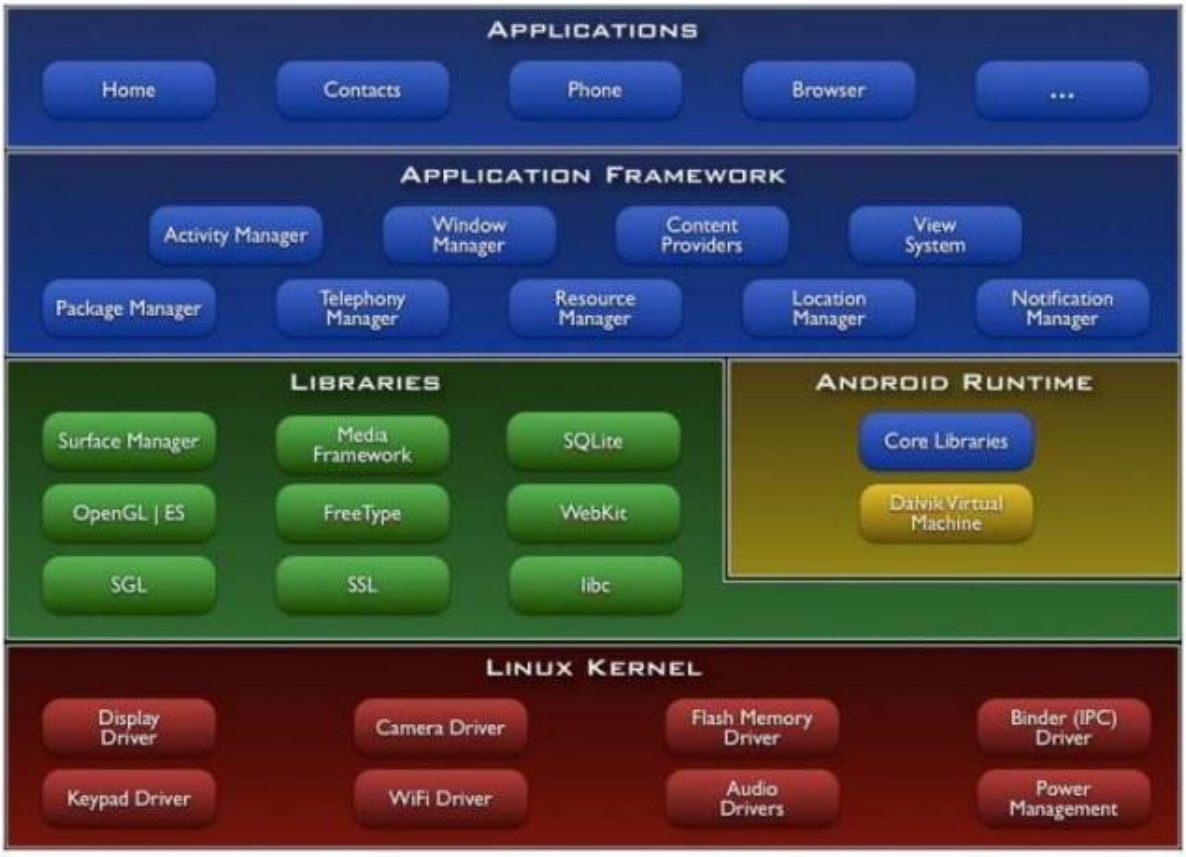

Figura 6: Camadas da Plataforma Android. Fonte [12].

3.2.1 Aplications

Acima de todas as outras camadas está a camada de aplicativos, na qual se encontram todos os aplicativos (escritos em Java) do Android, como cliente de email, navegador web, contatos e outros [12].

3.2.2 Application Framework

Logo abaixo da camada de aplicativos está a do framework. Nela encontra-se todas as APIs (Interfaces de Programação de Aplicativos) e recursos que os aplicativos utilizarão, como classes

visuais que implementam botões e views, provedor de conteúdo e gerenciadores de recursos, de notificação e de pacotes [12].

3.2.3 Libraries

Como o próprio nome diz, essa é a camada que possui as bibliotecas. Carrega consigo um conjunto de bibliotecas C/C++ utilizadas pelo sistema. Estão incluídas nesse conjunto a biblioteca C padrão (Libc) e também aquelas das áreas de multimídia, visualização de camadas 2D e 3D, funções para navegadores web, funções para gráficos, funções de aceleração de hardware, renderização 3D, fontes bitmap e vetorizadas e funções de acesso ao banco SQLite [12].

3.2.4 Android Runtime

É a camada do ambiente de execução. Ela é uma instância da máquina virtual Dalvik criada para cada aplicação executada no Android. A Dalvik é uma máquina virtual com melhor desempenho, maior integração com a nova geração de hardware e projetada para executar várias máquinas virtuais paralelamente. Além disso, é otimizada para consumo mínimo de memória, bateria e CPU [12].

A máquina virtual Dalvik conta também com o kernel do Linux para a funcionalidade base como *threading* e gerenciamento de memória de baixo nível [13].

3.2.5 Linux Kernel

A ultima camada é composta pelo kernel Linux na versão 2.6. Essa camada é a responsável pelo gerenciamento da memória e processos, gerir os serviços, redes e drivers, além de fazer uma abstração do hardware do dispositivo [12].

4. O JOGO

4.1 BREVE HISTÓRIA

Após sua longa viagem intercontinental Flyga, um mosquito-fêmea Aedes Aegypti, começou a invasão ao continente Americano espalhando sua terrível doença, a Dengue. Apesar de sua idade avançada, Flyga conseguiu gerar vários filhos e filhas que herdaram sua vontade e continuam o ataque aos humanos mesmo com o passar dos séculos. Entretanto, somente suas filhas eram capazes de transmitir tal doença e foi assim de geração a geração. Apesar de frágeis e pequenos, o mosquito possui uma capacidade muito grande de adaptar-se a novos lugares, o que dificulta a luta que parecia muito fácil para o ser humano vencer.

No ano de 2012 nasce Flya, que em pouco tempo aprende a lutar contra os humanos, mas ainda é muito desajeitada. Em sua primeira tentativa de "caça" acaba sendo perseguida por Bob, um Agente de Combate e se perde em um labirinto cheio de obstáculos.

4.2 PERSONAGENS PRINCIPAIS

4.2.1 Flya – Aedes Aegypti

Características

- Um mosquito-fêmea atrapalhado que gosta desafios.
- Sua comida preferida é sangue do tipo O-.
- Tem aproximadamente 0.5cm de comprimento.
- Seu hobby é colocar ovos em água parada.

4.2.2 Bob – Homo Sapiens Sapiens

Características

- Um Agente de Combate a Endemias que odeia mosquitos hematófagos.
- Sua comida preferida é pizza de calabresa.
- Tem 25 anos e mede 1,78m.
- Seu hobby é destruir possíveis criadouros do mosquito da dengue.

4.3 MOVIMENTAÇÃO

Toda movimentação é feita através das direcionais que aparecem no canto inferior esquerdo da tela. Observe as Figuras 5 e 6.

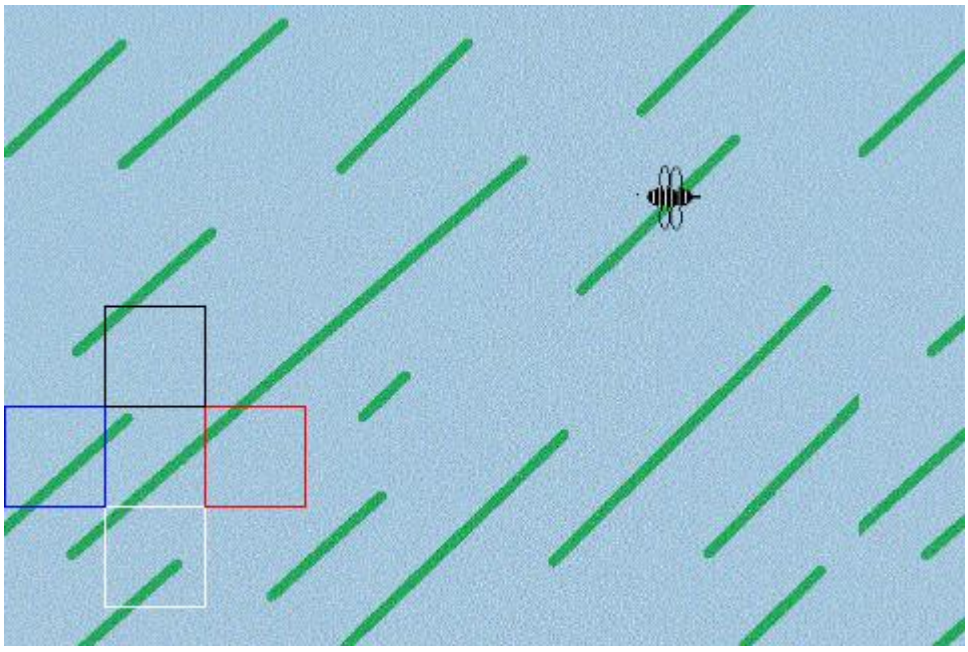

Figura 7: Mosquito movendo-se para direita.

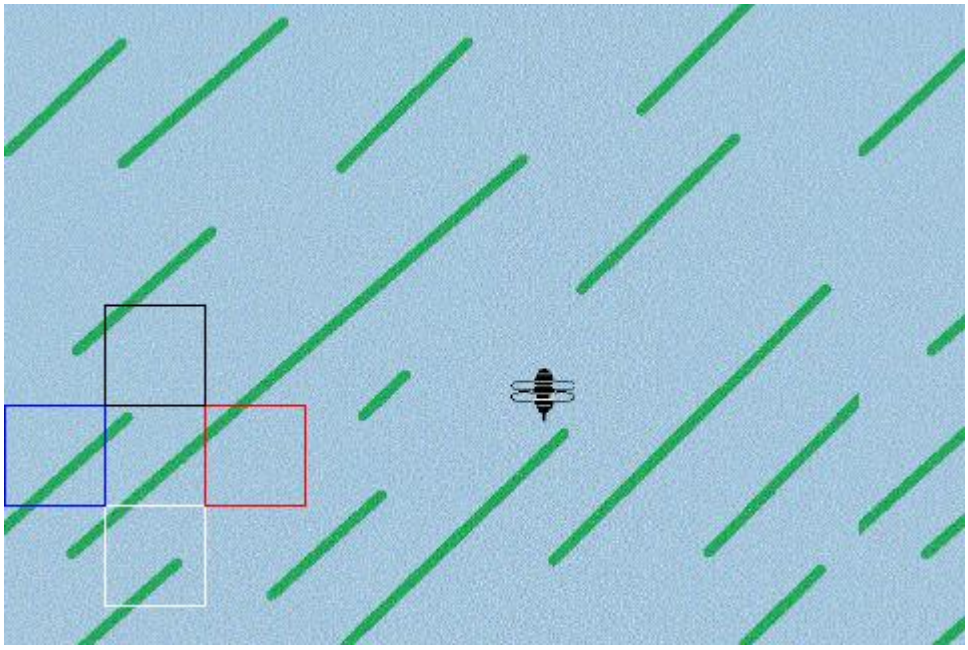

Figura 8: Mosquito movendo-se para baixo.

4.4 OBJETIVO

Neste jogo você ajudará Bob no combate a dengue, evitando deixar Flya colocar ovos em possíveis criadouros presentes no labirinto e respondendo corretamente a questões básicas sobre a doença.

5. MATERIAIS E MÉTODOS

Para o desenvolvimento do aplicativo será utilizado o IDE Eclipse junto ao plugin fornecido pela Google, o *Android Development Tools* (ADT), que auxilia no processo de desenvolvimento de um aplicativo para plataforma Android. Além disso, não será utilizada bibliotecas ou engines de terceiros, apenas bibliotecas disponíveis em Java e Android.

6. ANÁLISE E PROJETO DO JOGO

Neste capítulo estarão presentes todas as documentações do aplicativo desde os casos de uso até os diagramas de sequencia separados por atores.

6.1 BOB

6.1.1 Casos de Uso

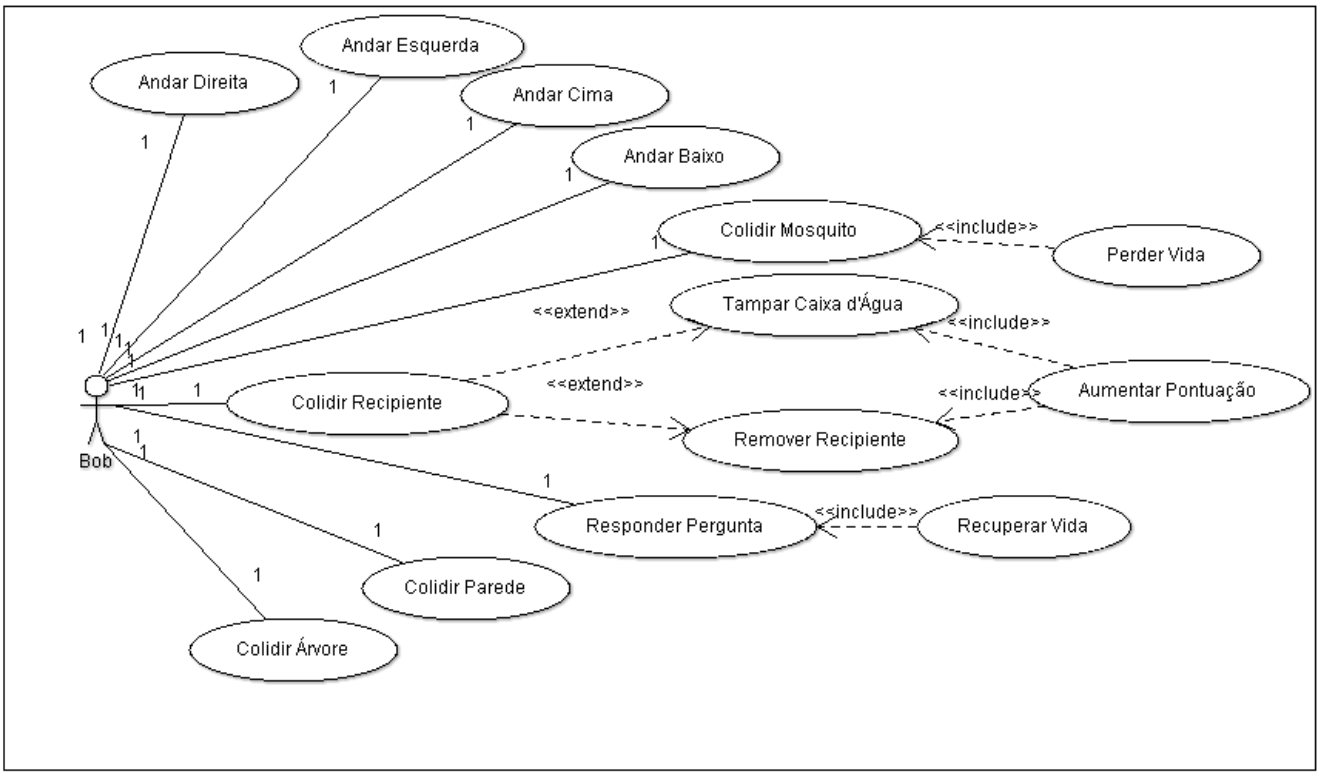

Figura 9 – Casos de Uso Bob.

6.1.2 Narrativa dos Casos de Uso

6.1.2.1 Andar Direita

1. Finalidade
    a. Movimentar Bob para direita.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. Iniciar alguma fase do jogo.
4. Fluxo Principal
    a. O Usuário toca no direcional que movimentará o Bob para direita.
    b. Bob movimenta-se para direita. (5a)(5b)
5. Fluxo Alternativo
    a. Colide com um recipiente a sua direita.
    b. Colide com o mosquito.
6. Fluxo Exceção
7. Testes
    a. O sistema verifica se foi tocado o direcional para direita.
    b. O sistema verifica se houve colisão com recipiente.
    c. O sistema verifica se houve colisão com mosquito.

6.1.2.2 Andar Esquerda

1. Finalidade
    a. Movimentar Bob para esquerda.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. Iniciar alguma fase do jogo.
4. Fluxo Principal
    a. O Usuário toca no direcional que movimentará o Bob para esquerda.
    b. Bob movimenta-se para esquerda. (5a)(5b)
5. Fluxo Alternativo
    a. Colide com um recipiente a sua esquerda.
    b. Colide com o mosquito.
6. Fluxo Exceção
7. Testes
    a. O sistema verifica se foi tocado o direcional para esquerda.
    b. O sistema verifica se houve colisão com recipiente.
    c. O sistema verifica se houve colisão com mosquito.

6.1.2.3 Andar Cima

1. Finalidade
    a. Movimentar Bob para cima.
2. Atores
    a. Bob.

3. Pré – Requisito
    a. Iniciar alguma fase do jogo.
4. Fluxo Principal
    a. O Usuário toca no direcional que movimentará o Bob para cima.
    b. Bob movimenta-se para cima. (5a)(5b)
5. Fluxo Alternativo
    a. Colide com um recipiente em cima dele.
    b. Colide com o mosquito.
6. Fluxo Exceção
7. Testes
    a. O sistema verifica se foi tocado o direcional para cima.
    b. O sistema verifica se houve colisão com recipiente.
    c. O sistema verifica se houve colisão com mosquito.

6.1.2.4 Andar Baixo

1. Finalidade
    a. Movimentar Bob para baixo.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. Iniciar alguma fase do jogo.
4. Fluxo Principal
    a. O Usuário toca no direcional que movimentará o Bob para baixo.

   b. Bob movimenta-se para baixo. (5a)(5b)

5. Fluxo Alternativo
   a. Colide com um recipiente em baixo dele.
   b. Colide com o mosquito.
6. Fluxo Exceção
7. Testes
   a. O sistema verifica se foi tocado o direcional para baixo.
   b. O sistema verifica se houve colisão com recipiente.
   c. O sistema verifica se houve colisão com mosquito.

6.1.2.5 Colidir Mosquito

1. Finalidade
   a. Fazer com que o jogador perca uma vida.
2. Atores
   a. Bob.
3. Pré – Requisito
   a. Iniciar alguma fase do jogo.
4. Fluxo Principal
   a. O Bob colide com um mosquito.
   b. Uma vida é retirada do Bob. (5.a)
5. Fluxo Alternativo
   a. Caso jogador possuir apenas uma vida a fase será finalizada.
6. Fluxo Exceção

7. Testes
    a. O sistema verifica se houve colisão com mosquito.
    b. O sistema verifica a quantidade de vidas.

6.1.2.6 Perder Vida

1. Finalidade
    a. Fazer com que o jogo tenha uma dificuldade maior.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. Bob colidir com o mosquito.
4. Fluxo Principal
    a. O Bob colide com um mosquito.
    b. Uma vida é retirada do Bob.(5.a)
5. Fluxo Alternativo
    a. Caso jogador possuir apenas uma vida a fase será finalizada.
6. Fluxo Exceção
7. Testes
    a. O sistema verifica a quantidade de vidas.

6.1.2.7 Colidir Recipiente

1. Finalidade

   a. Demonstrar que estes recipientes são possíveis criadouros do mosquito da dengue numa situação real.

2. Atores
   a. Bob.
3. Pré – Requisito
   a. Bob colidir com algum recipiente.
4. Fluxo Principal
   a. O Bob colide com um recipiente.
   b. É feito o controle mecânico sobre o recipiente colidido.(5.a)(5.b)
   c. Aumenta 10 pontos na pontuação geral.
5. Fluxo Alternativo
   a. Caso o recipiente colidido seja comum ele será removido da tela.
   b. Caso o recipiente colidido seja uma caixa d'água, será tampada.
6. Fluxo Exceção
7. Testes
   a. O sistema verifica se houve colisão com recipiente comum.
   b. O sistema verifica se houve colisão com uma caixa d'água.

6.1.2.8 Remover Recipiente

1. Finalidade
   a. Fazer com que o Usuário ganhe pontos e combata o mosquito.
2. Atores
   a. Bob.

3. Pré – Requisito
    a. Bob colidir com recipiente comum.
4. Fluxo Principal
    a. O Bob colide com um recipiente comum.
    b. É feito o controle mecânico sobre o recipiente colidido.
    c. Aumenta 10 pontos na pontuação geral.
5. Fluxo Alternativo
6. Fluxo Exceção
7. Testes

6.1.2.9 Tampar Caixa D'Água

1. Finalidade
    a. Fazer com que o Usuário ganhe pontos e combata o mosquito.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. Bob colidir com caixa d'água.
4. Fluxo Principal
    a. O Bob colide com uma caixa d'água.
    b. É feito o controle mecânico sobre o recipiente colidido.
    c. Aumenta 10 pontos na pontuação geral.
5. Fluxo Alternativo

6. Fluxo Exceção
7. Testes

6.1.2.10 Aumentar Pontuação

1. Finalidade
    a. Dar dinâmica ao jogo.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. Bob colidir com algum recipiente.
4. Fluxo Principal
    a. O sistema adiciona mais 10 pontos à pontuação geral.
5. Fluxo Alternativo
6. Fluxo Exceção
7. Testes

6.1.2.11 Responder Pergunta

1. Finalidade
    a. Dar dinâmica ao jogo e passar informações sobre a dengue.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. O Jogador ter apenas uma vida
4. Fluxo Principal
    a. O sistema abre a tela de pergunta
    b. O jogador seleciona uma opção (5.a)
    c. O sistema aumenta uma vida do jogador.
5. Fluxo Alternativo
    a. O jogador erra a resposta
6. Fluxo Exceção
7. Testes
    a. O sistema verifica a quantidade de vida do jogador.
    b. O sistema verifica a resposta do jogador.

6.1.2.12 Recuperar Vida

1. Finalidade
    a. Dar dinâmica ao jogo.

2. Atores
    a. Bob.
3. Pré – Requisito
    a. O Jogador ter respondido à pergunta corretamente.
4. Fluxo Principal
    a. O sistema adiciona uma vida ao jogador.
5. Fluxo Alternativo
6. Fluxo Exceção
7. Testes

6.1.2.13 Colidir Parede

1. Finalidade
    a. Bloquear passagem.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. O Jogador estar em movimento.
4. Fluxo Principal
    a. O sistema impede o jogador de seguir adiante.
5. Fluxo Alternativo
6. Fluxo Exceção
7. Testes
    a. O sistema verifica se houve a colisão com uma parede.

6.1.2.14 Colidir Árvore

1. Finalidade
    a. Bloquear passagem.
2. Atores
    a. Bob.
3. Pré – Requisito
    a. O Jogador estar em movimento.
4. Fluxo Principal
    a. O sistema impede o jogador de seguir adiante.
5. Fluxo Alternativo
6. Fluxo Exceção
7. Testes
    a. O sistema verifica se houve a colisão com uma árvore.

6.2 MOSQUITO

6.2.1 Casos de Uso

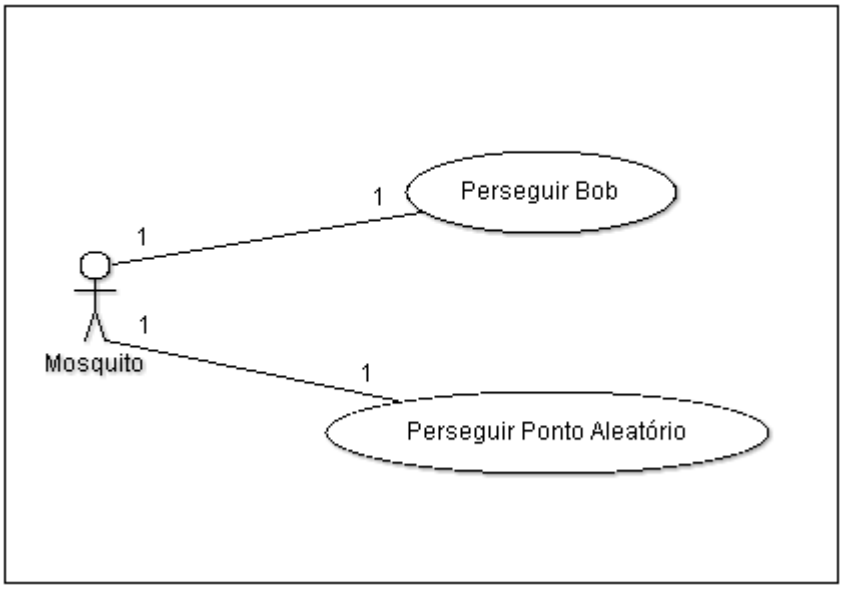

Figura 10 – Casos de Uso Mosquito.

6.2.2 Narrativas dos Casos de Uso

6.2.2.1 Perseguir Bob

1. Finalidade
    a. Dar dinâmica ao jogo.
2. Atores
    a. Mosquito.
3. Pré – Requisito
    a. Ter iniciado alguma fase.
4. Fluxo Principal
    a. O mosquito movimenta-se em direção ao Bob.(5.a)
5. Fluxo Alternativo

a. O mosquito movimenta-se em direção à um ponto aleatório.

6. Fluxo Exceção
7. Testes
    a. O sistema verifica a posição do Bob.

6.2.2.2 Perseguir Ponto Aleatório

1. Finalidade
    a. Dar dinâmica ao jogo.
2. Atores
    a. Mosquito.
3. Pré – Requisito
    a. Ter iniciado alguma fase.
4. Fluxo Principal
    a. O mosquito movimenta-se em direção à um ponto aleatório.(5.a)
5. Fluxo Alternativo
    a. O mosquito movimenta-se em direção ao Bob.
6. Fluxo Exceção
7. Testes
    a. O sistema seleciona um ponto aleatório.

6.3 USUÁRIO

6.3.1 Casos de Uso Usuário

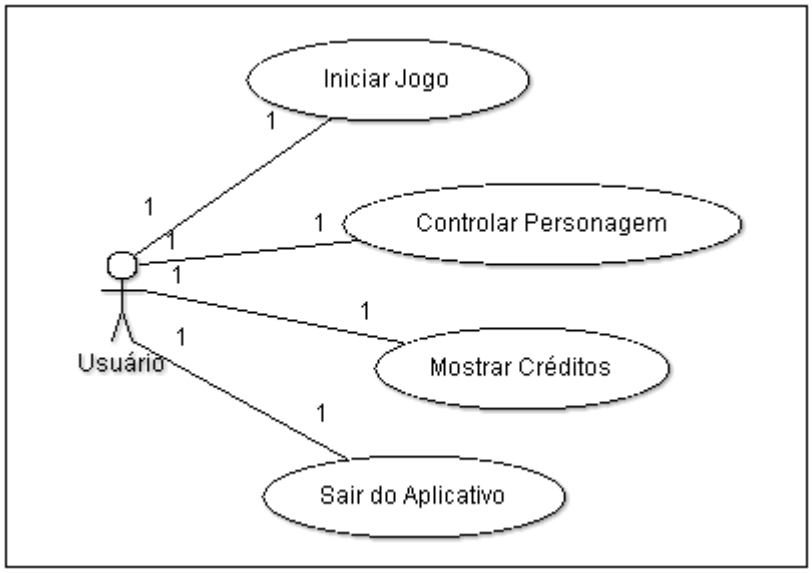

Figura 11 – Casos de Uso Usuário.

6.3.2 Narrativas dos Casos de Uso

6.3.2.1 Iniciar Jogo

1. Finalidade
    a. Dar início à primeira fase.
2. Atores
    a. Usuário.
3. Pré – Requisito
    a. Ter iniciado o aplicativo.

4. Fluxo Principal
    a. O usuário seleciona a opção Iniciar Jogo no menu principal.(5.a)(5.b)
5. Fluxo Alternativo
    a. O usuário seleciona Mostrar Créditos
    b. O usuário seleciona Sair do Jogo.
6. Fluxo Exceção
7. Testes
    a. O sistema verifica a opção selecionada no menu principal.

6.3.2.2 Controlar Personagem

1. Finalidade
    a. Mover Personagem.
2. Atores
    a. Usuário.
3. Pré – Requisito
    a. Ter iniciado o jogo.
4. Fluxo Principal
    a. O usuário pressiona uma das direções. (5.a)
5. Fluxo Alternativo
    a. O usuário não pressiona nenhuma das direções
6. Fluxo Exceção
7. Testes
    a. O sistema verifica se alguma direção foi pressionada.

6.3.2.3 Mostrar Créditos

1. Finalidade
    a. Demonstrar as informações sobre o aplicativo.
2. Atores
    a. Usuário.
3. Pré – Requisito
    a. Ter iniciado o aplicativo.
4. Fluxo Principal
    a. O usuário seleciona mostrar créditos no menu inicial.(5.a)(5.b)
5. Fluxo Alternativo
    a. O usuário seleciona a opção Iniciar Jogo.
    b. O usuário seleciona Sair do Jogo.
6. Fluxo Exceção
7. Testes
    a. O sistema verifica a opção selecionada pelo usuário.

6.3.2.4 Sair do Aplicativo

1. Finalidade
    a. Finalizar o aplicativo.
2. Atores
    a. Usuário.

3. Pré – Requisito
    a. Ter iniciado o aplicativo.
4. Fluxo Principal
    a. O usuário seleciona sair do jogo no menu inicial.(5.a)(5.b)
5. Fluxo Alternativo
    a. O usuário seleciona a opção Iniciar Jogo.
    b. O usuário seleciona Mostrar Créditos.
6. Fluxo Exceção
7. Testes
    a. O sistema verifica a opção selecionada pelo usuário.

6.4 APLICATIVO

6.4.1 Casos de Uso Aplicativo

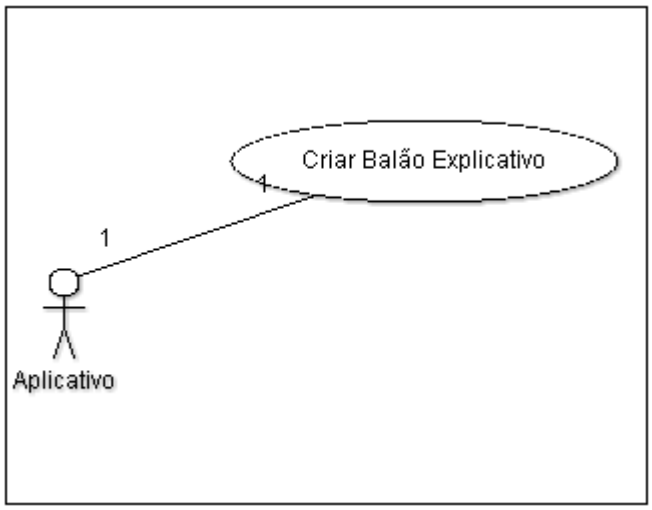

Figura 12 – Casos de Uso Aplicativo.

6.4.2 Narrativas dos Casos de Uso

6.4.2.1 Criar Balão Explicativo

1. Finalidade
    a. Criar o uma mensagem explicando coisas sobre o jogo.
2. Atores
    a. Aplicativo.
3. Pré – Requisito
    a. Remover algum tipo de recipiente pela primeira vez.
4. Fluxo Principal
    a. Personagem colide com um recipiente pela primeira vez.
    b. Balão explicativo é criado.
    c. Ao tocar na tela o balão será fechado.
5. Fluxo Alternativo
6. Fluxo Exceção
7. Testes

6.5 DIAGRAMAS DE SEQUÊNCIA

6.5.1 Andar Direita

6.5.1.1 Melhor Caso

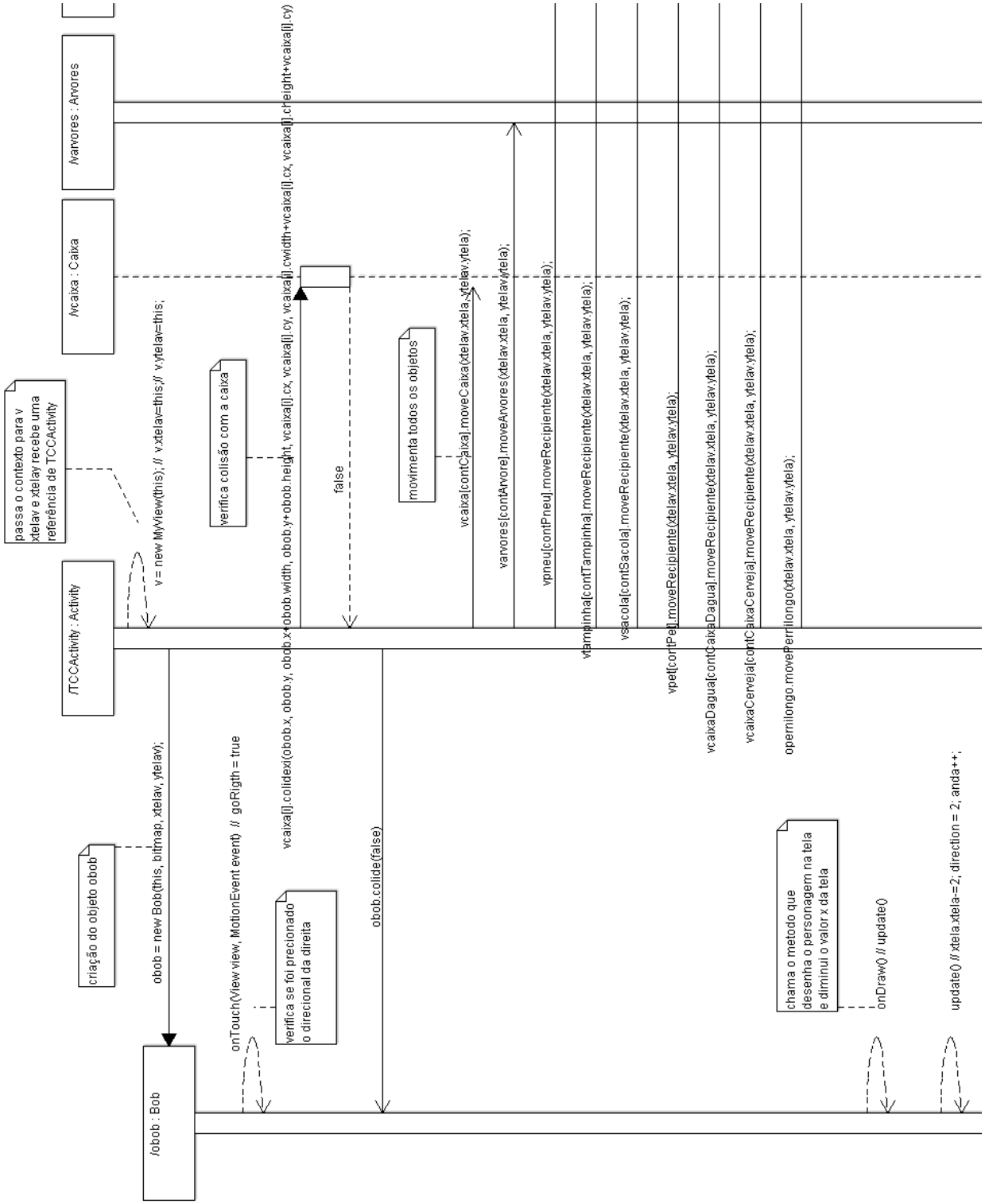

Figura 13 – Andar Direita Melhor Caso parte 1.

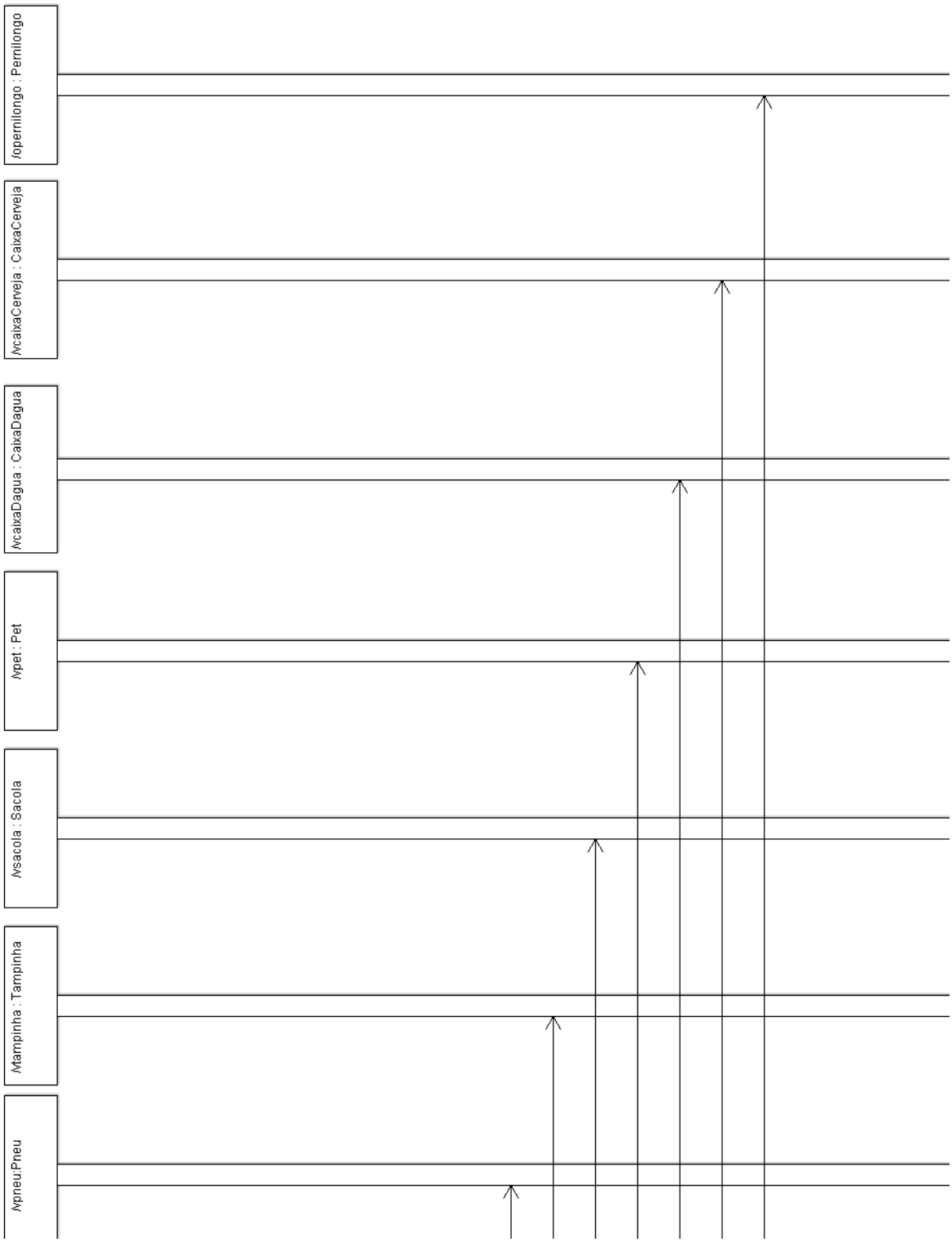

Figura 14 – Andar Direita Melhor Caso parte 2.

6.5.1.2 Pior Caso

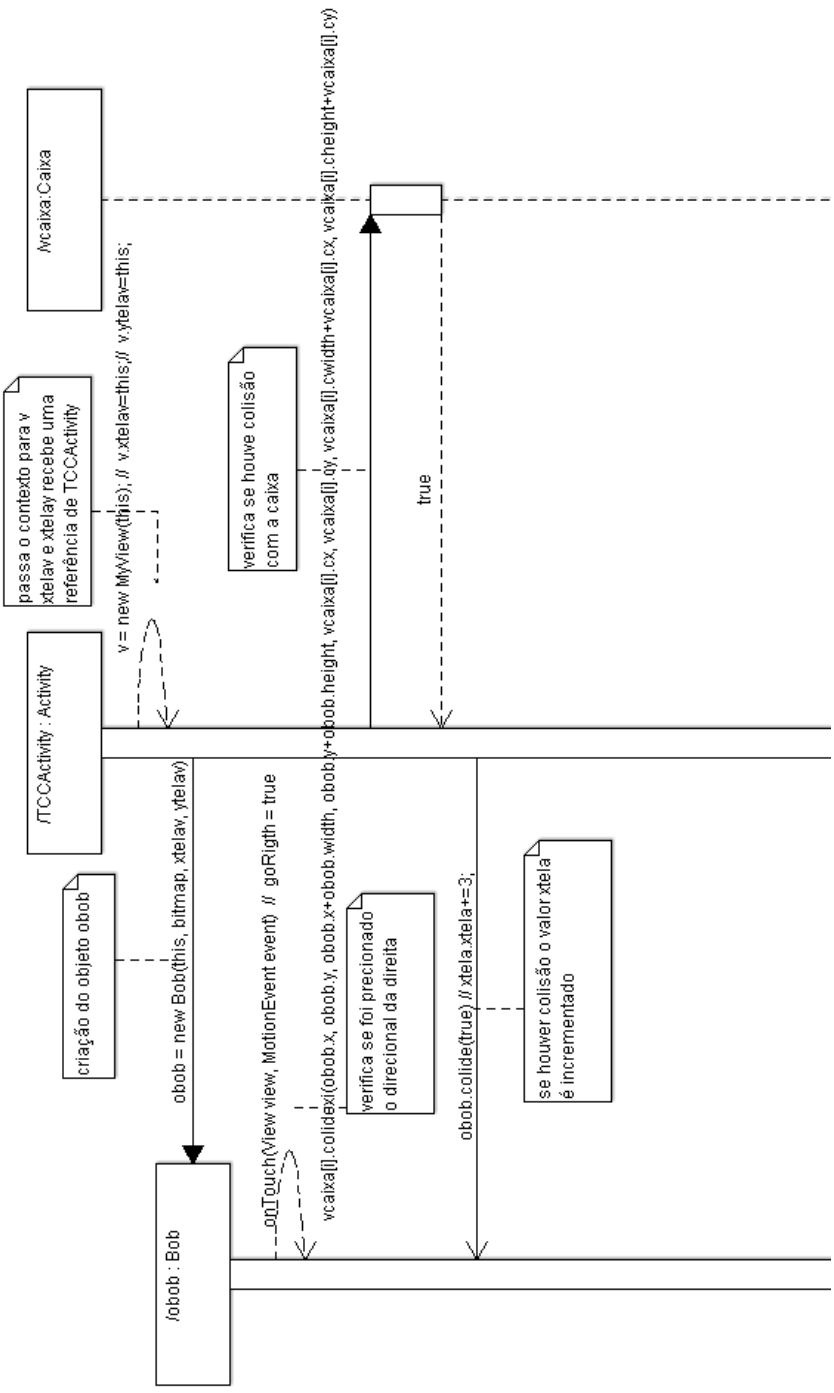

Figura 15 – Andar Direita Pior Caso.

6.5.2 Colidir Mosquito / Perder Vida

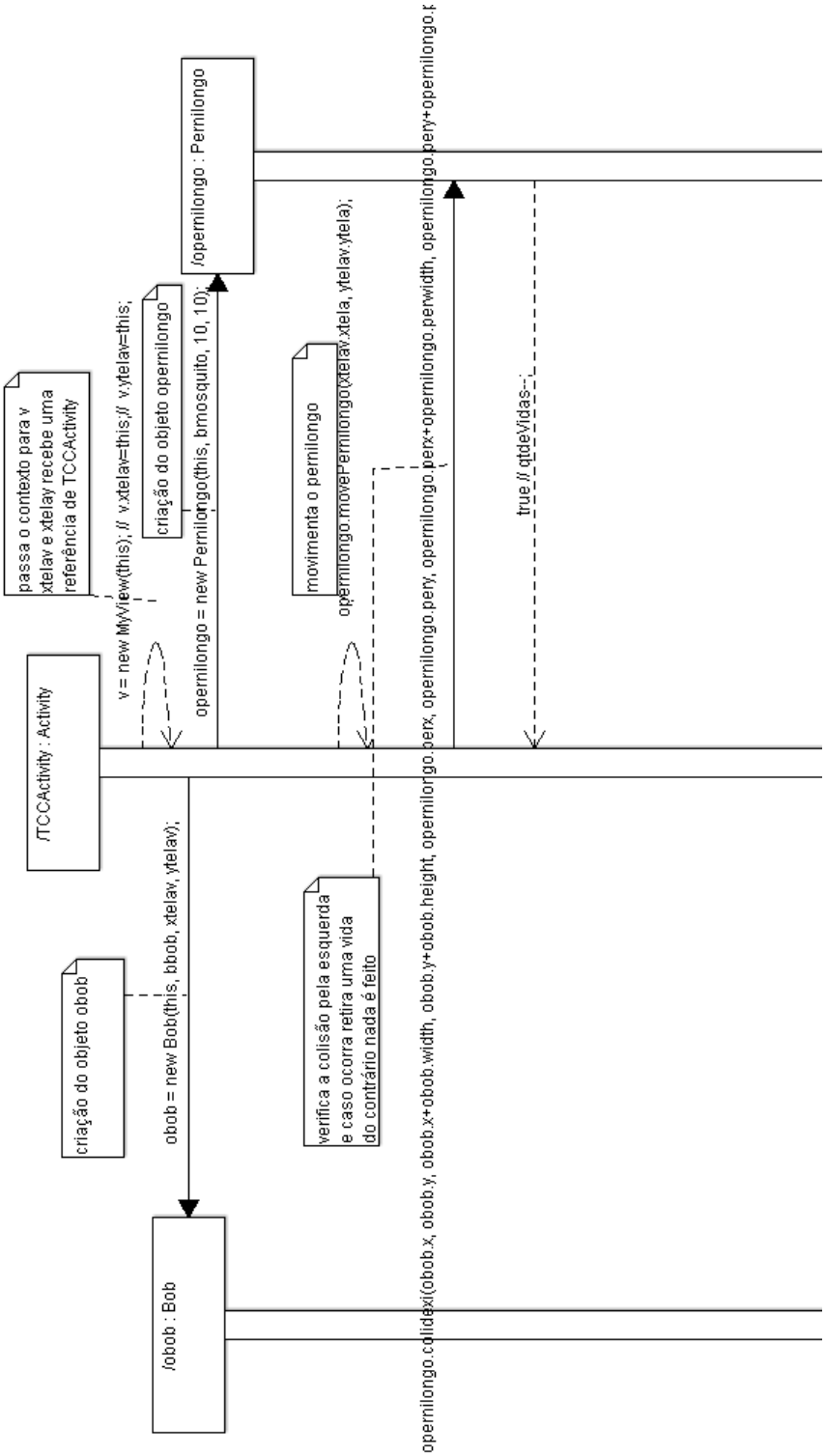

Figura 16 – Colidir Mosquito / Perder Vida.

6.5.3 Colidir Recipiente / Tampar Caixa D'água / Aumentar Pontuação

6.5.3.1 Melhor Caso

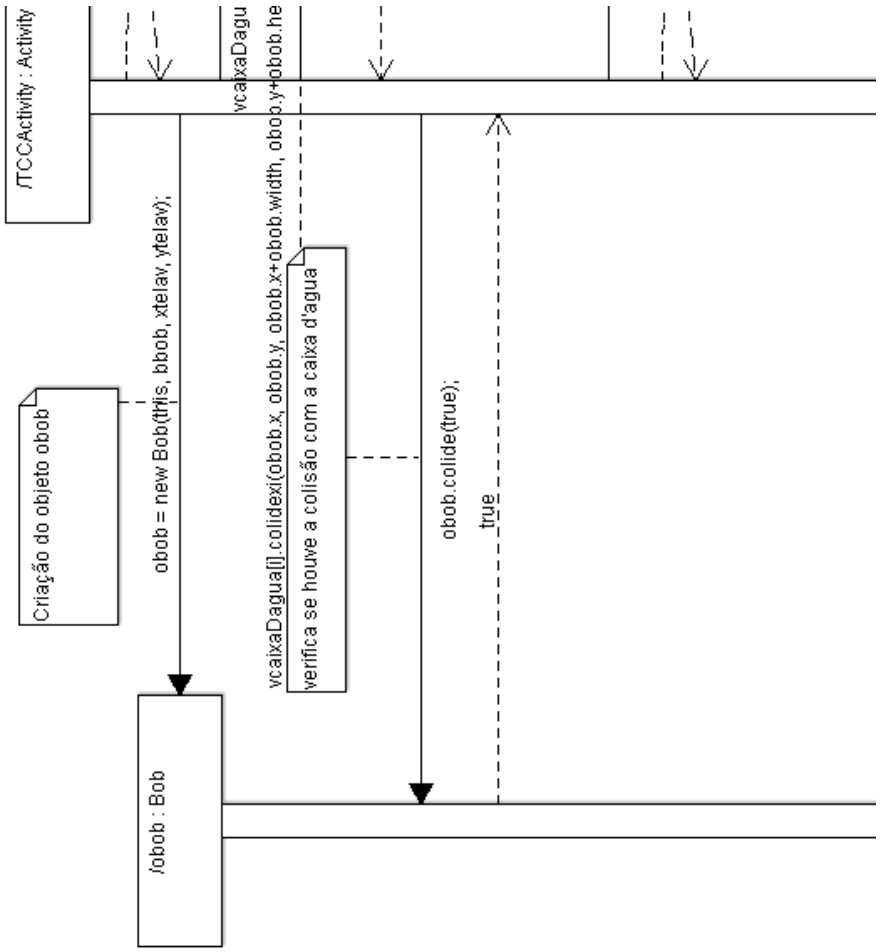

Figura 17 – Colidir Recipiente / Tampar Caixa D'água / Aumentar Pontuação Melhor Caso parte 1.

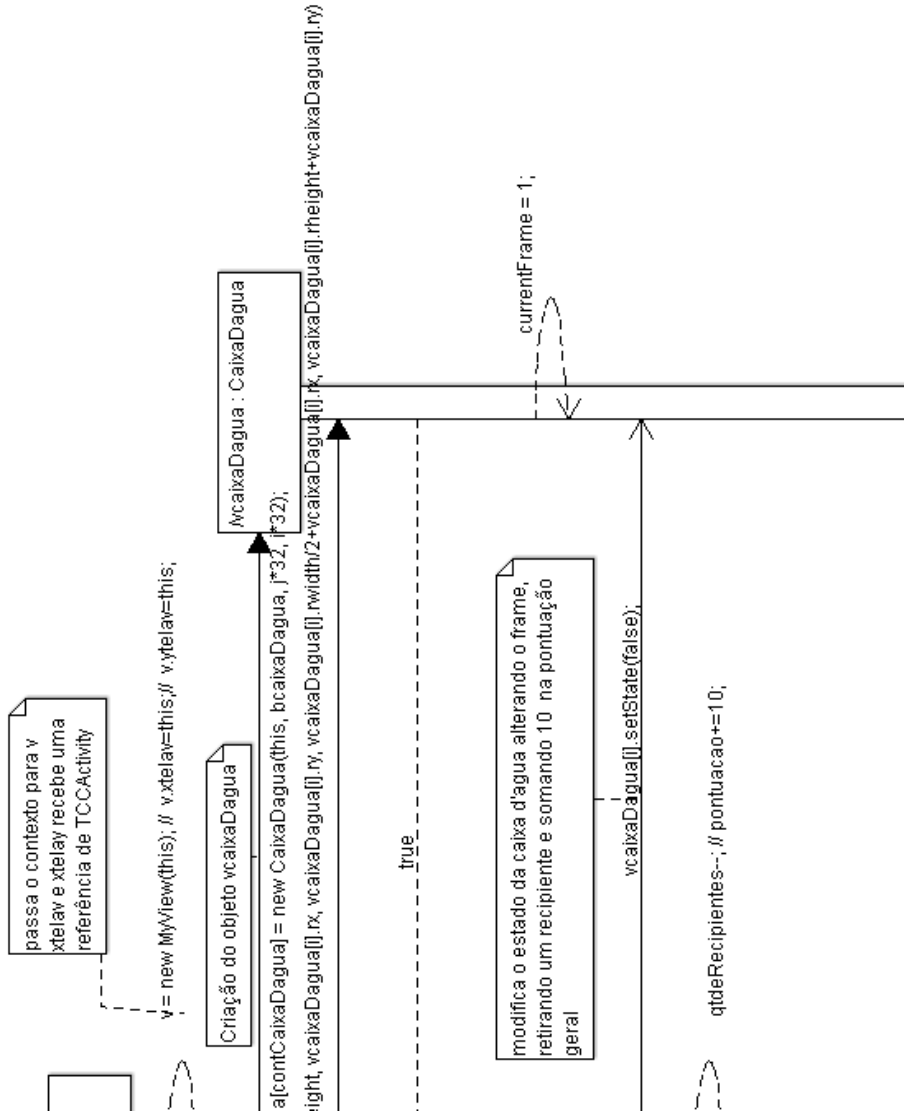

Figura 18 – Colidir Recipiente / Tampar Caixa D'água / Aumentar Pontuação Melhor Caso parte 2.

6.5.3.2 Pior Caso

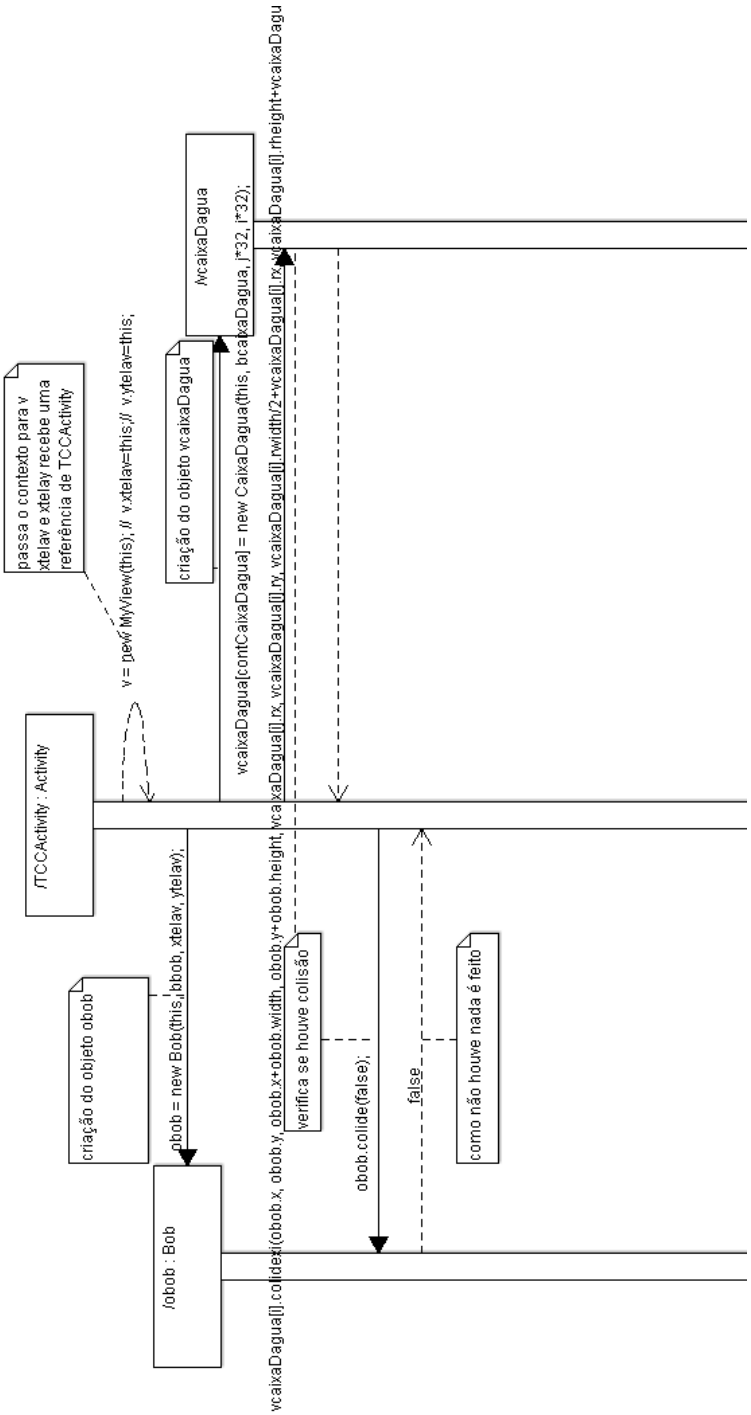

Figura 19 – Colidir Recipiente / Tampar Caixa D'água / Aumentar Pontuação Pior Caso.

6.5.4 Colidir Recipiente / Remover Recipiente / Aumentar Pontuação

6.5.4.1 Melhor Caso

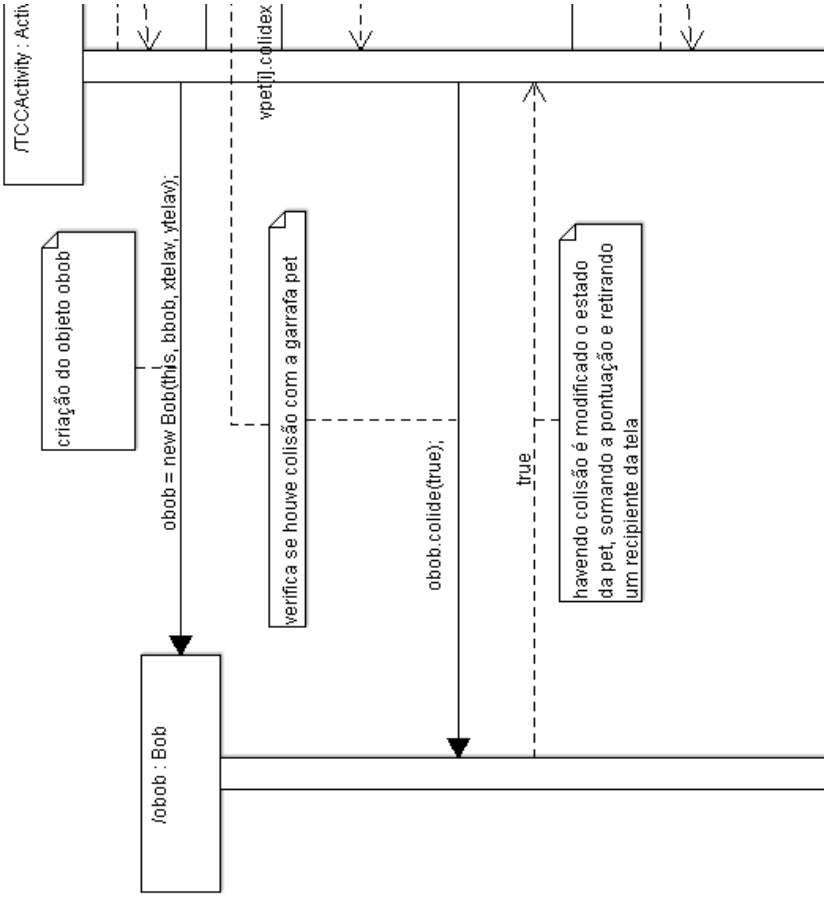

Figura 20 – Colidir Recipiente / Remover Recipiente / Aumentar Pontuação Melhor Caso parte 1.

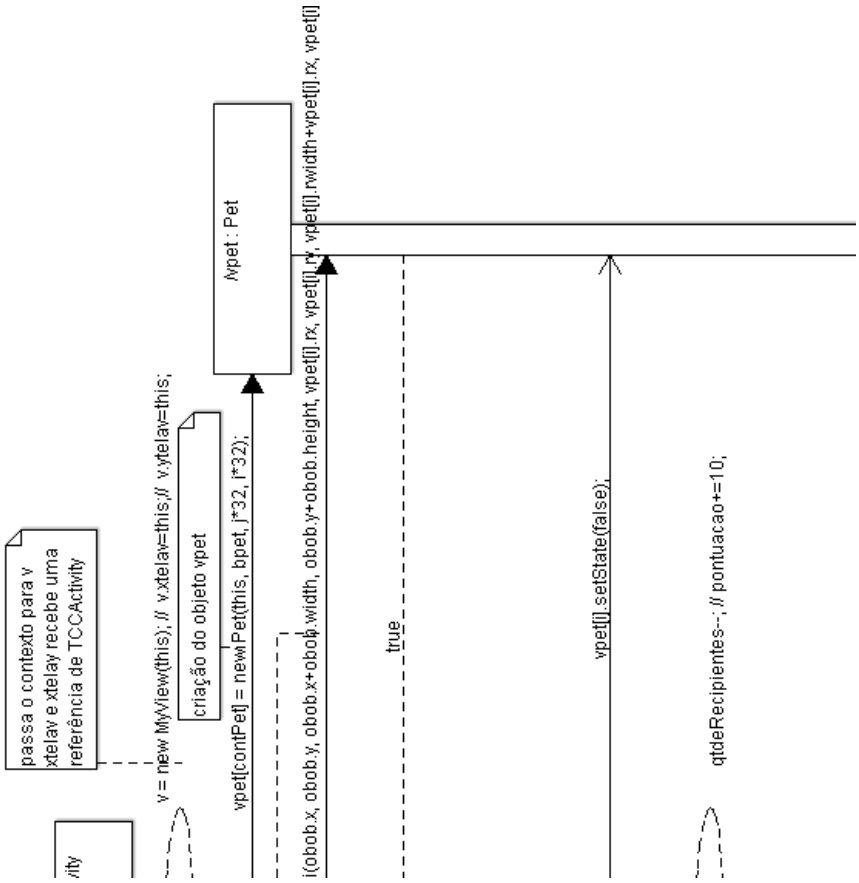

Figura 21 – Colidir Recipiente / Remover Recipiente / Aumentar Pontuação Melhor Caso parte 2.

6.5.4.2 Pior Caso

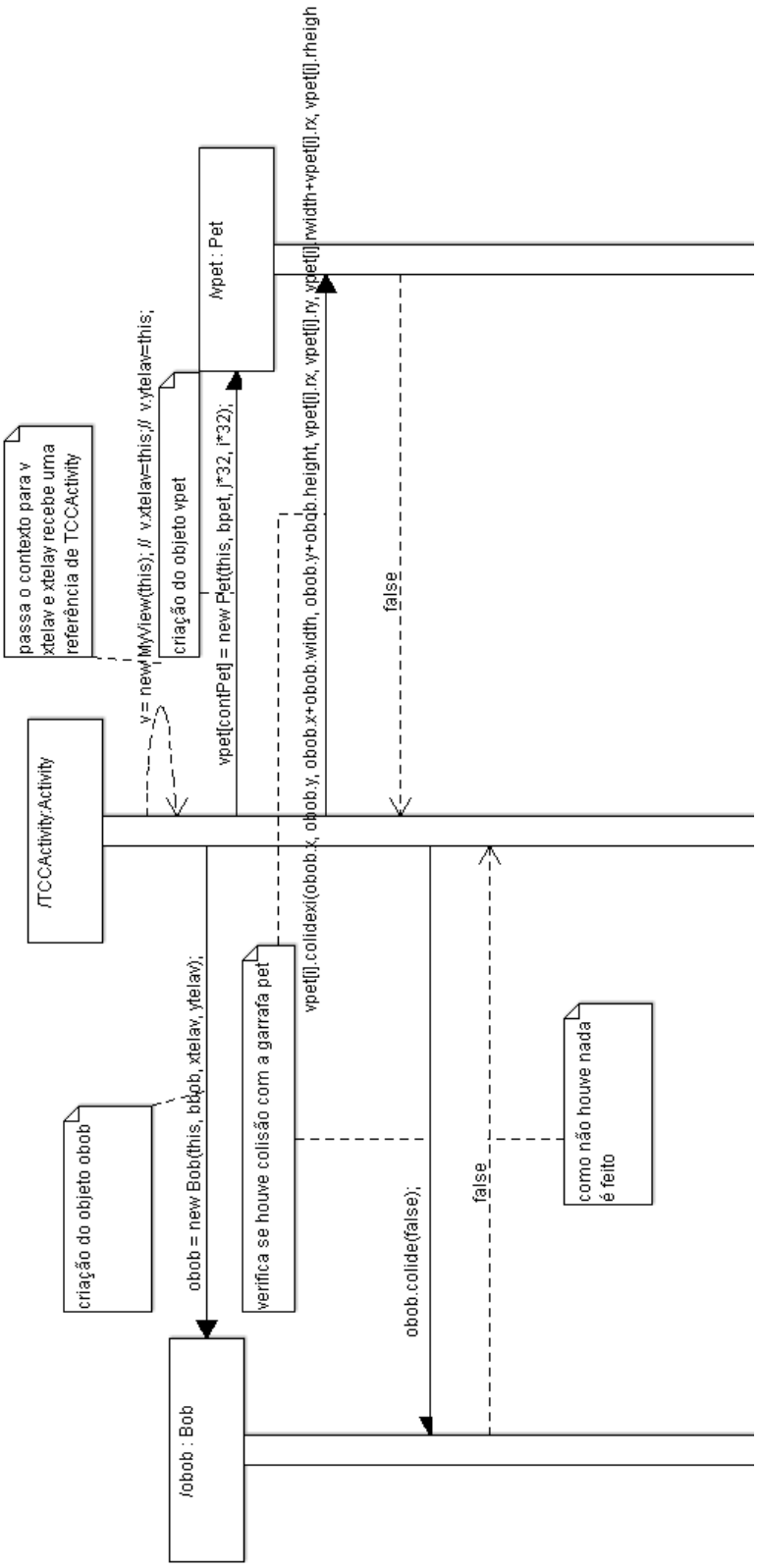

Figura 22 – Colidir Recipiente / Remover Recipiente / Aumentar Pontuação Pior Caso.

6.5.5 Responder Pergunta / Aumentar Vida

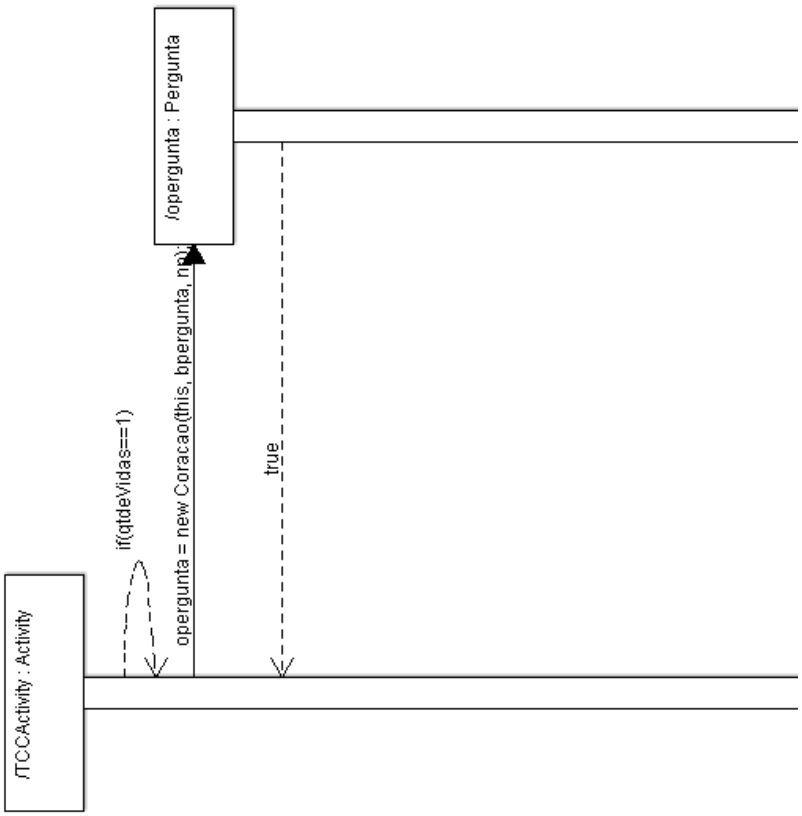

Figura 23 – Responder Pergunta / Aumentar Vida.

6.5.4 Colidir Parede / Colidir Árvore

6.5.6.1 Melhor Caso

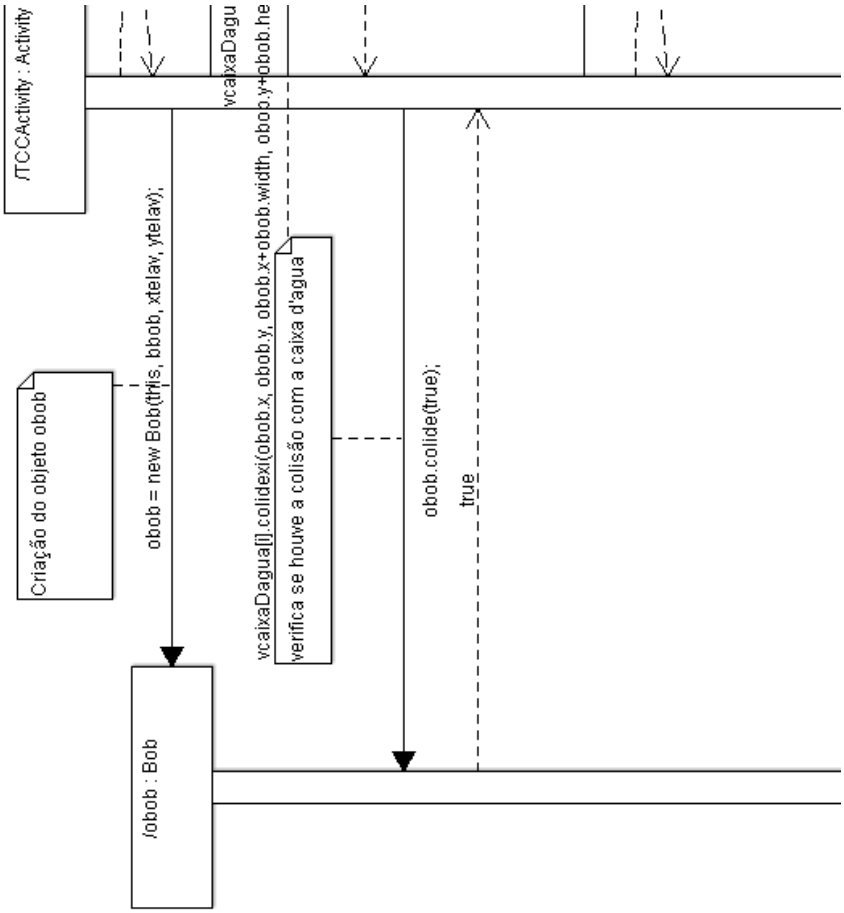

Figura 24 – Colidir Parede / Colidir Árvore Melhor Caso parte 1.

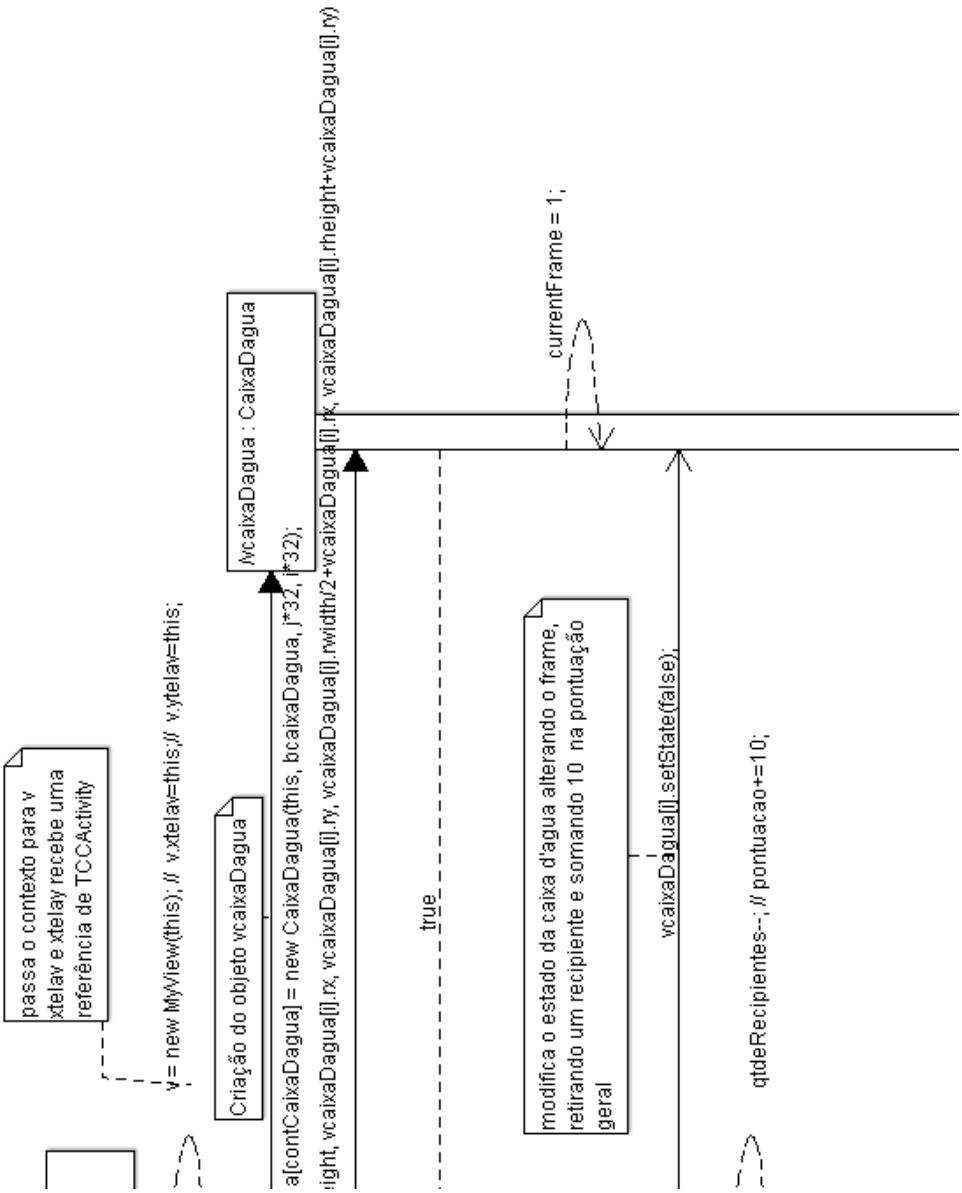

Figura 25 – Colidir Parede / Colidir Árvore Melhor Caso parte 2.

6.5.6.2 Pior Caso

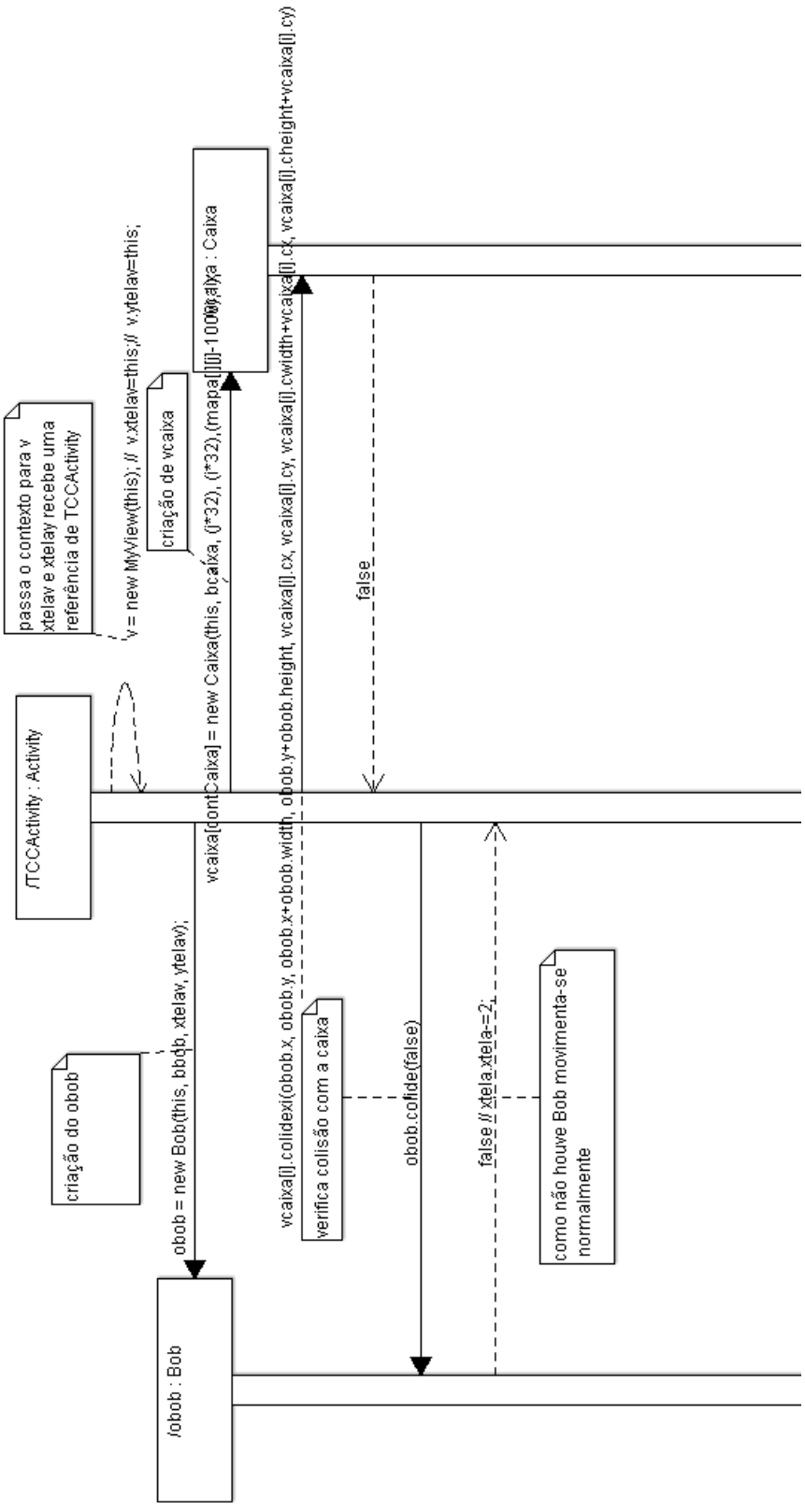

Figura 26 – Colidir Parede / Colidir Árvore Pior Caso.

6.5.5 Perseguir Bob

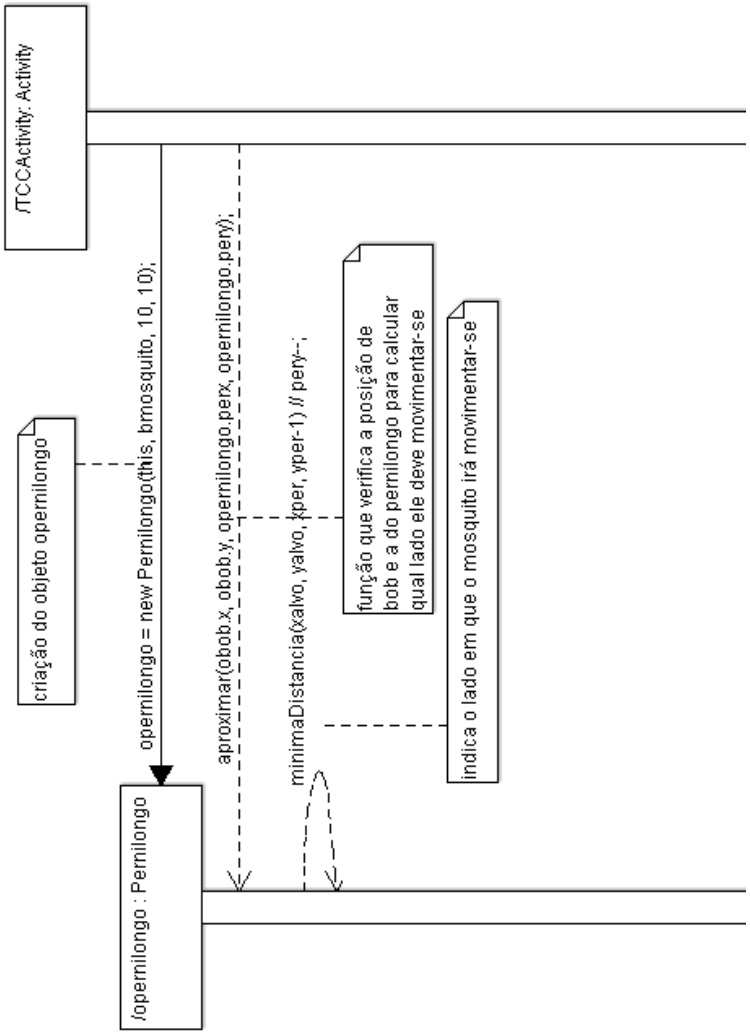

Figura 27 – Perseguir Bob.

6.5.6 Perseguir Ponto Aleatório

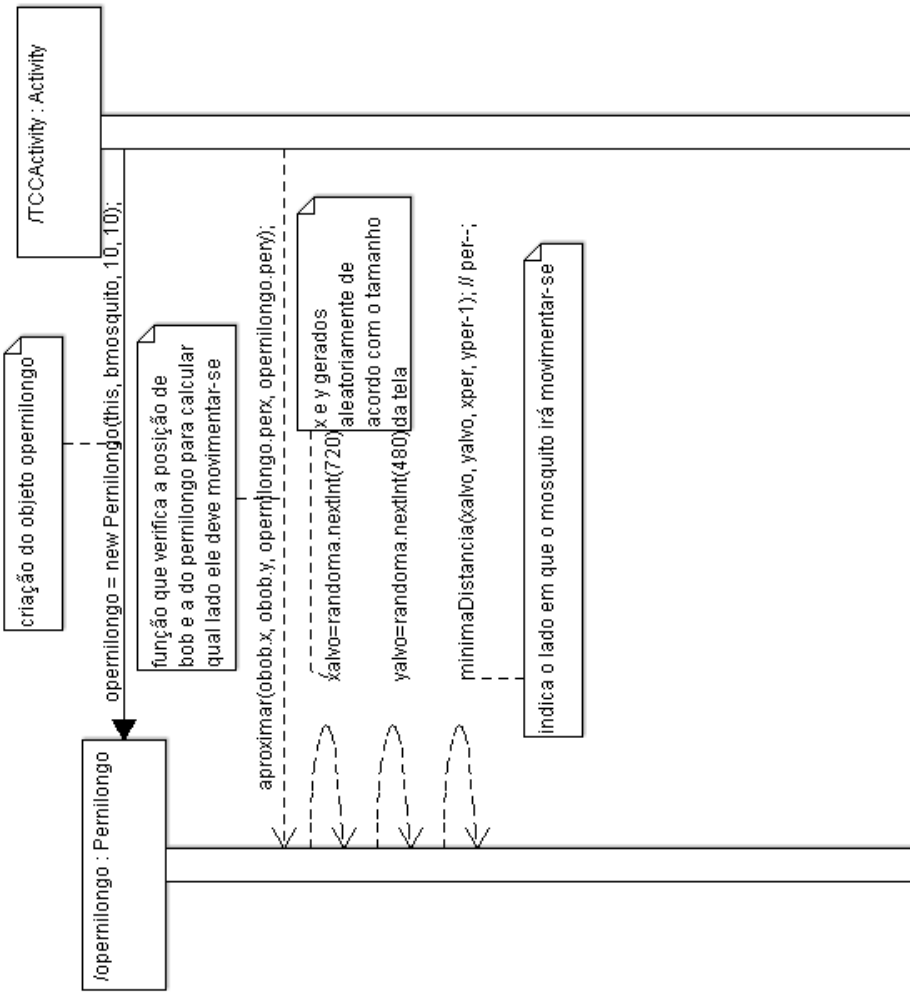

Figura 28 – Perseguir Ponto Aleatório.

6.5.7 Controlar Personagem

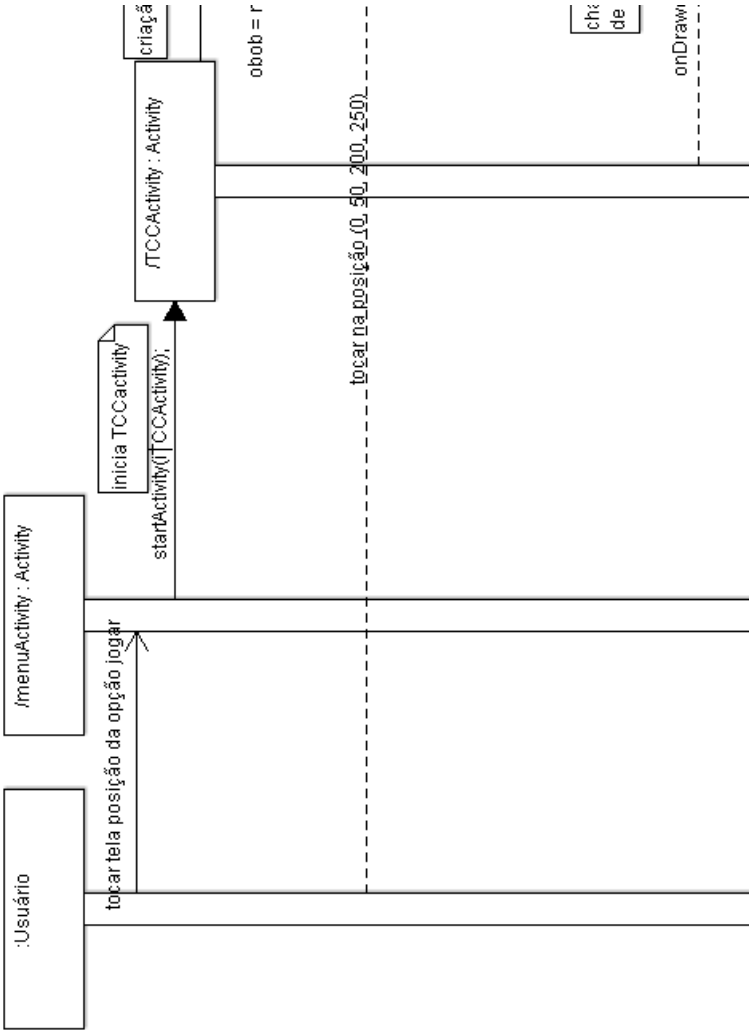

Figura 29 – Controlar Personagem parte 1.

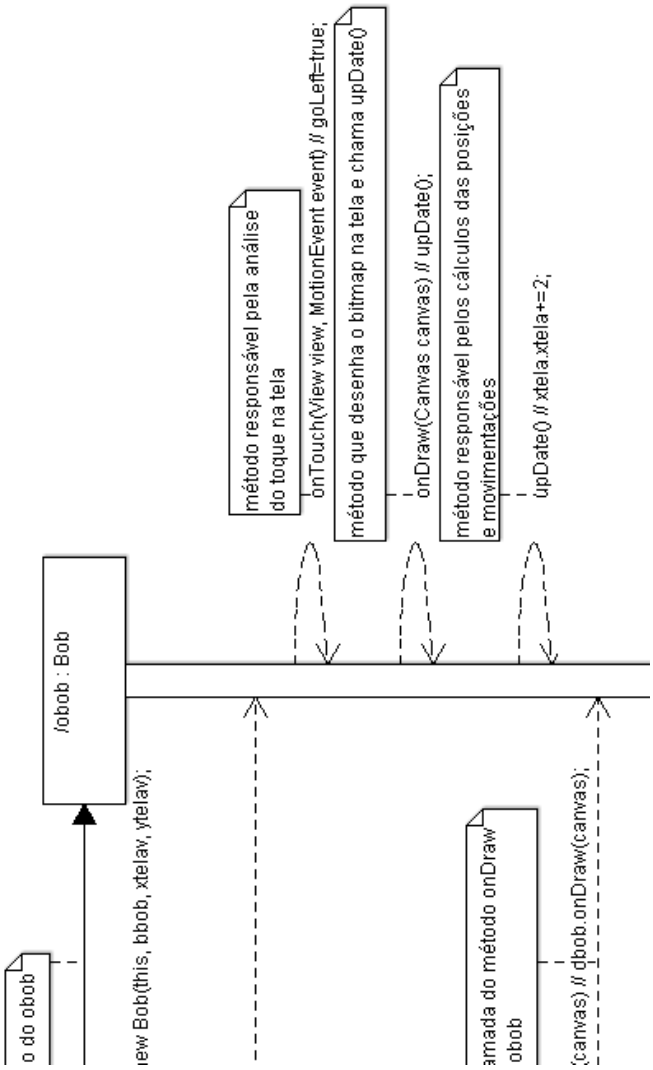

Figura 30 – Controlar Personagem parte 2.

6.5.8 Criar Balão Explicativo

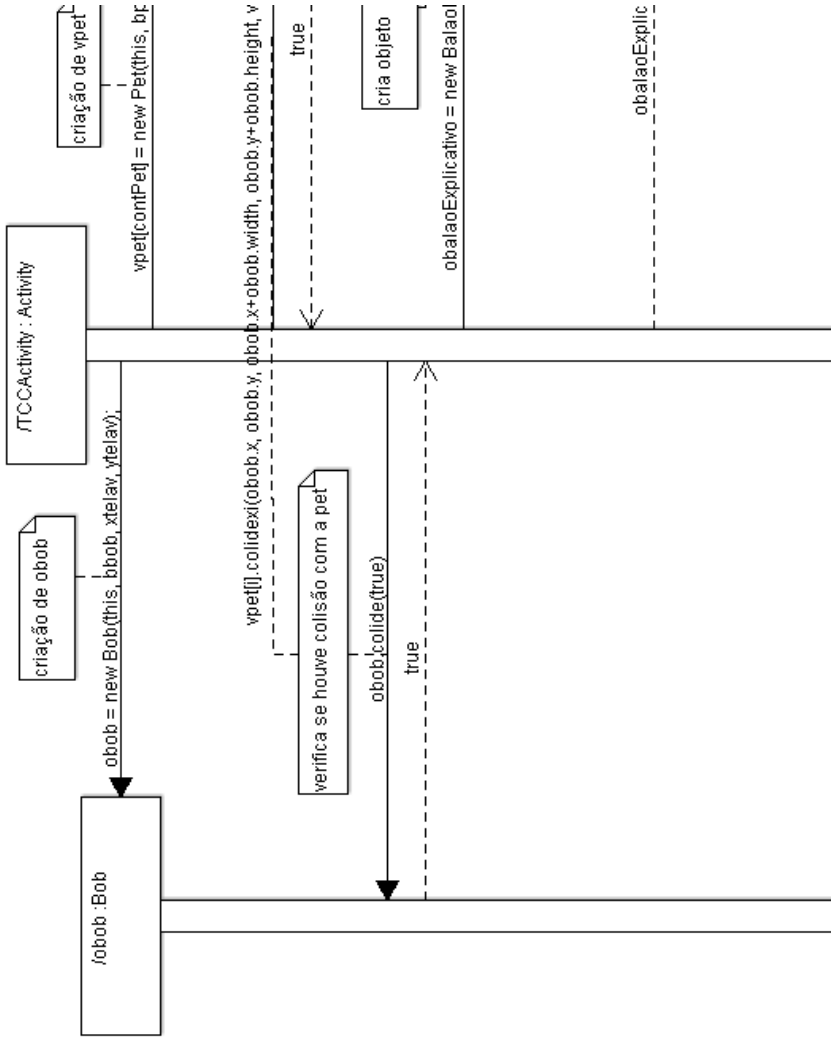

Figura 31 – Criar Balão Explicativo parte 1.

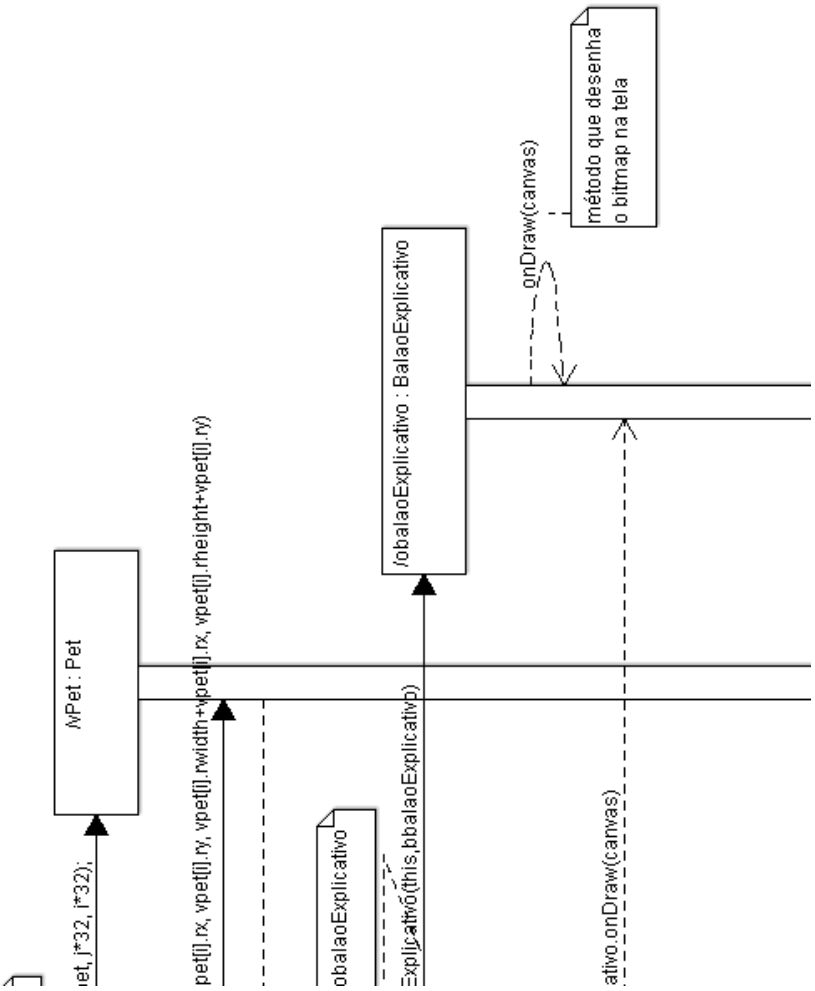

Figura 32 – Criar Balão Explicativo parte 2.

6.6 DIAGRAMA DE CLASSES

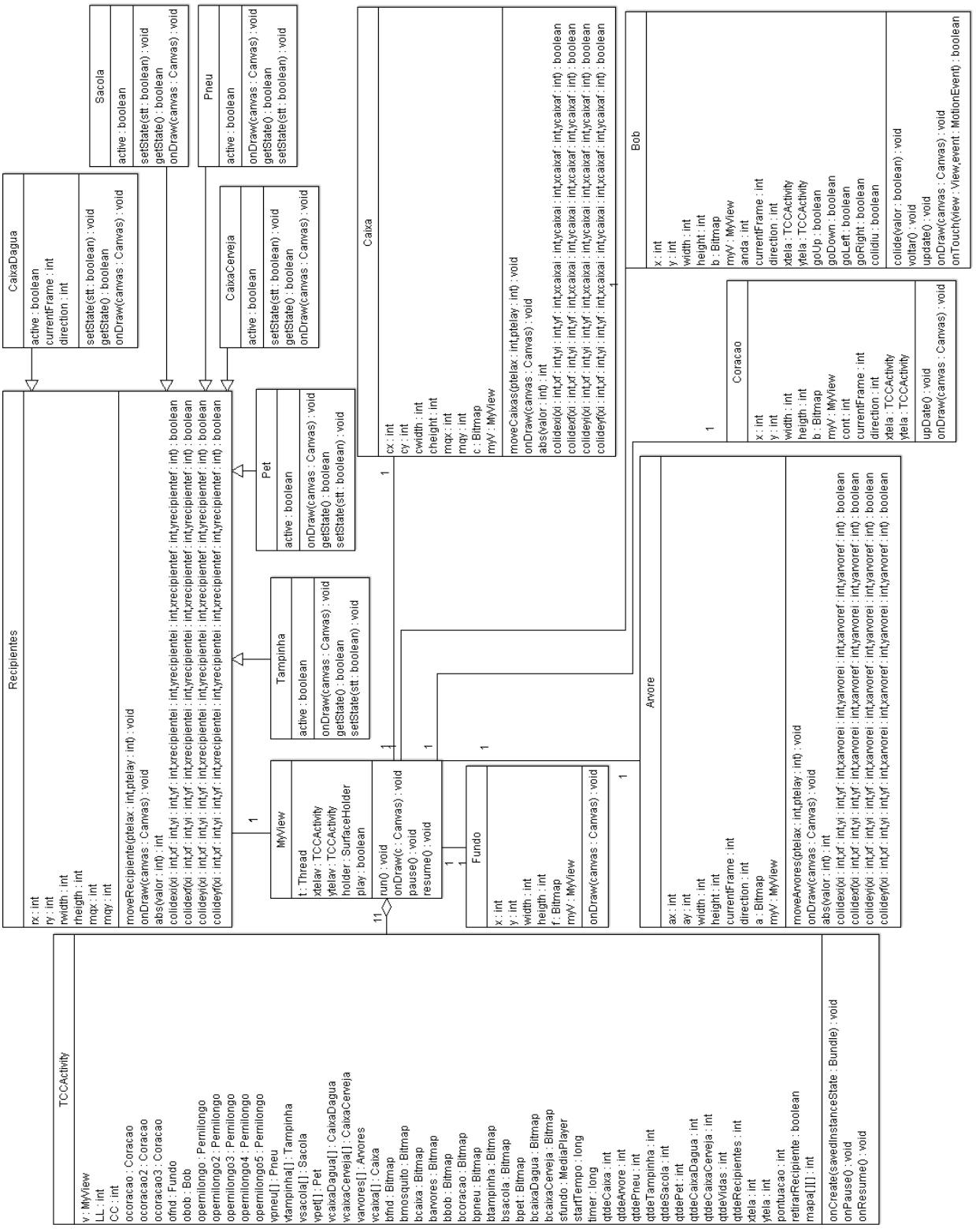

Figura 33 – Diagrama de Classes.

7. RESULTADOS

Aqui serão listadas algumas das ações e componentes presentes no jogo com seu respectivo código fonte.

7.1 MOVIMENTAÇÃO DO PERSONAGEM PRINCIPAL

No código abaixo será mostrado a implementação da classe Bob, a classe do personagem principal. É mostrado também como utilizar a interface OnTouchListener que é responsável pelo controle dos eventos que podem ser disparados com o toque na tela do dispositivo. O mesmo aplica-se para movimentação para direita, esquerda e para cima presentes no método upDate().

```
public class Bob implements OnTouchListener{
//atributos de Bob
int x, y;
static TCCActivity xtela, ytela;
//variaveis para verificar o lado pressionado
boolean goUp=false, goDown=false, goLeft=false, goRight=false, colidiu=false;
public Bob(MyView myView, Bitmap bob, TCCActivity xtelav, TCCActivity ytelav) {
      xtela = xtelav; //construtor
      ytela = ytelav;
      myV = myView;
      myV.setOnTouchListener(this); //define o ontouchlistener
      height = bob.getHeight() / 4;   //define o tamanho y da sprite
      width = bob.getWidth() / 4; //define o tamanho x da sprite
      x = 240-16; //posição inicial do bob
      y = 160-16;
      }
private void upDate() {
            if(!colidiu){
            if(goDown){
                  if(y+b.getHeight() < myV.getHeight()){
                                          //verifica se não bateu no canto da tela
                        ytela.ytela-=2;    //movimenta tela para baixo
                        goUp = goRight = goLeft = false;
                                                }
                     } }     }
public void onDraw(Canvas canvas) {
            upDate();
            //controle dos frames da sprite
            int srcX = currentFrame * width;
            int srcY = direction * height;
            Rect src = new Rect(srcX, srcY, srcX + width , srcY + height);
            Rect dst = new Rect(x, y, x+width, y+height);
            //desenha na tela
            canvas.drawBitmap(b, src, dst, null);
            …
}
}
```

Figura 34 - Bob Movendo para Cima.

Figura 35 - Bob Movendo para Direita.

7.2 COLISÃO COM GARRAFA PET

Aqui ocorre a colisão com um recipiente, no caso uma garrafa pet. A lógica funciona da seguinte forma: o objeto recipiente verifica se houve colisão de Bob com ele. Se houve, avisa Bob através do método colide do mesmo, alterando o estado de colidiu. Verifica-se em seguida este mesmo estado, e se o mesmo se encontrar em verdadeiro, o recipiente é removido do local através da alteração do método setState do mesmo.

```
public class TCCActivity extends Activity {
// declaração de variáveis e objetos que serão utilizados no jogo
final int LL = 30,CC = 30;      //linhas e colunas na matriz mapa
MyView v;                                                    //visão
Bob obob;                                                    //agente de endemias
Pet[] vpet = new Pet[10]; //numero máximo de garrafas pet
Bitmap bbob, bpet;
int xtela=0, ytela=0, pontuacao = 0;
public void onCreate(Bundle savedInstanceState) {
v = new MyView(this);     //passando o contexto para o objeto da clase MyView
        v.xtelav=this;
        v.ytelav=this;    //definindo x e y de v para movimentacao da tela
      //atribuindo os bitmaps as imagens do diretorio
      bbob = BitmapFactory.decodeResource(getResources(), R.drawable.bob);
      bpet = BitmapFactory.decodeResource(getResources(), R.drawable.pet);
       //chama a visão v
       setContentView(v);
}
public class MyView extends SurfaceView implements Runnable{
             ...
             TCCActivity xtelav, ytelav;
 public void run() {
      obob = new Bob(this, bbob, xtelav, ytelav); //criando objeto obob
      //criando Bob e garrafas pet
      for(int i = 0; i < CC; i++)
            for(int j =0; j<LL;j++){
                  if(mapa[i][j] == 6){    //
                        vpet[contPet] = new Pet(this, bpet, j*32, i*32);
                                    }       }   ...
      while(play){ ...
            for(int i = 0; i < CC; i++)            //movimenta as garrafas Pet
                  for(int j =0; j < LL;j++){
                        else if(mapa[i][j] == 6){
                        vpet[contPet].moveRecipiente(xtelav.xtela, ytelav.ytela);
                        contPet++;    }    } ...
                 }
            for(int i =0 ; i<qtdePet; i++){ //verificando colisões do bob com pet
                  if(vpet[i].getState()){
                  obob.colide(vpet[i].colidexi(obob.x, obob.y, obob.x+obob.width,
obob.y+obob.height, vpet[i].rx, vpet[i].ry, vpet[i].rwidth+vpet[i].rx,
vpet[i].rheight+vpet[i].ry));// colisao pela esquerda
                  if(obob.colidiu){
                  vpet[i].setState(false); //retirando garrafa pet
                                    }}       ...
                                       }}
```

Neste trecho de código é possível verificar a declaração de algumas das variáveis utilizadas no aplicativo e como ele é dividido, além da implementação da classe MyView que é a responsável pela criação, verificação de testes e impressão dos objetos na tela. E por ultimo um teste que verifica se houve a colisão com a garrafa pet somente pela esquerda.

Figura 36 – Bob Colide com as Garrafas Pet.

Figura 37– Bob Recolhe as Garrafas Pet.

7.3 MOVIMENTAÇÃO DO PERNILONGO

```
public class TCCActivity extends Activity {
Bob obob;                                    //agente de endemias
Pernilongo opernilongo, opernilongo2;        //mosquito da dengue
...
public void onCreate(Bundle savedInstanceState) {
//atribuindo os bitmaps as imagens do diretorio
bbob = BitmapFactory.decodeResource(getResources(), R.drawable.bob);
bmosquito = BitmapFactory.decodeResource(getResources(), R.drawable.mosquito);
...
}
public class MyView extends SurfaceView implements Runnable{
       ...
public void run() {
       obob = new Bob(this, bbob, xtelav, ytelav);
       opernilongo = new Pernilongo(this, bmosquito, 10, 10);
       while(play){
       opernilongo.movePernilongo(xtelav.xtela, ytelav.ytela);
       opernilongo.aproximar(obob.x, obob.y, opernilongo.perx, opernilongo.pery);
       ...
                  }
       ...
                   }

}
}
```

Acima se pode observar a declaração dos objetos das classes Bob e Pernilongo assim como atribuição de suas respectivas imagens e sua criação no método run(). Os métodos movePernilongo(int, int) e aproximar(int, int, int, int) são os responsáveis pela movimentação do pernilongo e perseguição do Bob ou um ponto aleatório.

```
public class Pernilongo{
int perx, pery;
...                         //atributos
int escolhas[];
boolean estilo = false;
int xalvo, yalvo, tempoEstilo=400;
Random randoma;
//construtor
public Pernilongo(MyView myView, Bitmap pernilongo, int x, int y) {
            per = pernilongo;
            myV = myView;
            perx = x; //posição inicial do pernilongo
            pery = y;
            ...
            randoma = new Random();
            if(randoma.nextInt(2)==0)estilo=false; //define o estilo
            else estilo=true;                     //bob ou aleatorio
}
//método para aproximar pernilongo do Bob ou
//algum objeto aleatório de acordo com estilo
public void aproximar(int xalvo, int yalvo, int xper, int yper){
            escolhas = new int [4];
            tempoEstilo--; //qtde de iterações de um pernilongo em um estilo
            if(tempoEstilo<1){ //se acabou o tempo sorteia um novo estilo
                  if(randoma.nextInt(2)==0)estilo=false;
                  else estilo=true;
                  tempoEstilo=400;} //redefine tempo
            if(estilo){               //estilo define se o alvo será bob ou
                  xalvo=this.xalvo;   //um ponto aleatório
                  yalvo=this.yalvo;
                  if(perx==xalvo && pery==yalvo){
                        this.xalvo=randoma.nextInt(720);
                        this.yalvo=randoma.nextInt(480);
                  }
            }   //calcula mínima distancia para Bob ou alvo
      escolhas[0] = minimaDistancia(xalvo, yalvo, xper, yper-1);
            ...
            for(int i = 1; i < 4; i++){
                  if(escolhas[i]<escolhas[i-1]){
                        menorValor = escolhas[i];
                        indice = i;
                  }
            }
            if(indice == 0){
                  pery--;
                  currentFrame=0;
            }
            ...
}  //calculo da distancia manhatam
public int minimaDistancia(int xalvo, int yalvo, int xper, int yper){
            return (abs(xalvo-xper)+abs(yalvo-yper));
      }
public void movePernilongo(int ptelax, int ptelay){ //move o pernilongo junto
com a tela
            perx = perx+ptelax;
            pery = pery+ptelay;
      }
...
}
```

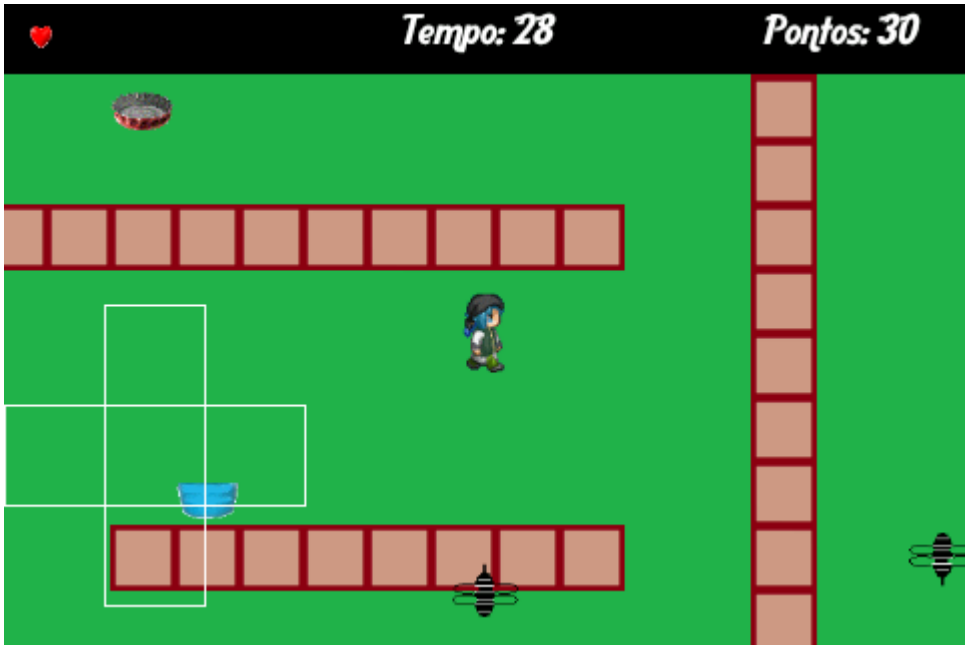

Figura 38 – Mosquitos se Movimentando.

7.4 MATRIZ DO MAPA

Matriz utilizada para definição do mapa com seus respectivos valores.

```
public class TCCActivity extends Activity {
// declaração de variáveis e objetos que serão utilizados no jogo
...
/*=============================================================================*|
|                                  matriz mapa onde                            |
|  2 = arvore // 3 = pneu // 4 = tampinha // 5 = sacola // 6 = pet //          |
|  7 = caixadagua // 8 = caixacerveja                                          |
|  >1000 = linha de caixas na horizontal e >2000 = linha de caixas na vertical  |
*=============================================================================*/

int mapa[][]=//30x30
       {{2030,1028,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2030},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,0,0,2,2008,7,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,1003,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,2,0,0,0,0,5,0},
       {0,0,0,0,0,2016,1008,0,0,0,0,0,0,0,2008,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,2004,2004,6,6,6,0,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,2,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,0,0,0,0},
       {0,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,2003,2003,0,0,0,0,0,0,1010,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,2006,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,8,0,0,0,7,0,0,0,0,0,0,0,0,0,0,0,0,0,7,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,1006,0,0,0,0,0,0,0,0,0,0,0,0,1008,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,1006,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,2,2,6,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,6,0,0,0},
       {0,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,2,2,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,0,0,2,2,0,0,0,0},
       {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0},
       {0,5,0,0,0,0,0,0,0,5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0},
       {0,1028,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}};

...

}
```

8. CONCLUSÃO

Com tudo apresentado até agora, é possível perceber que o Android é a nova tendência do momento e o desenvolvimento de aplicativos para esta plataforma está em alta. Além de que atualmente é difícil encontrar pessoas que nunca tiveram contato com um dispositivo móvel, independente da idade.

Isso torna visível que a produção de jogos educativos para a plataforma Android, ou outras plataformas também, é um mercado que deve ser explorado, pois nele é possível combinar duas coisas que quando aliadas corretamente só produzem bons resultados: a conscientização e o entretenimento.

Combinar o lúdico com conhecimento tem por objetivo ensinar sem que o usuário perca o foco. Espera-se com isto a fixação e conscientização de diversas camadas da população.

Como desafio pessoal, o trabalho apresentado serviu como base e desafio suficiente para um crescimento da capacidade de trabalho do autor. Houve uma combinação interdisciplinar de áreas, visando à construção de uma ferramenta computacional, utilizando uma tecnologia em evidência (conhecimento da tecnologia), ensinando sobre a dengue (conhecimento sobre a doença), para um público leigo.

9. REFERÊNCIAS

[1] HALSTEAD, Scott B.. Dengue: Tropical Medicine: Science and Practice. Londres: Imperial College Press, 2008. 485 p.

[2] Site da Dengue. Mantido pela AJA Brasil (Associação do Jovem Aprendiz). Uma organização não governamental, sem fins lucrativos, não religiosa, apartidária e independente. Disponível em: < http://www.dengue.org.br/index.html >. Acesso em: 10 junho 2012.

[3] BENSEÑOR, Isabela. "HowStuffWorks - Como funciona a dengue". Publicado em 09 de maio de 2007 (atualizado em 09 de maio de 2008). Disponível em: <http://saude.hsw.uol.com.br/dengue.htm>. Acesso em: 05 junho 2012.

[4] WHITE, Katherine. Dengue Fever,1. ed. Nova Iorque: Editora Rosen Publishing, 2004.

[5] MINISTÉRIO DA SAÚDE. O agente comunitário de saúde no controle da dengue. 1.ed. Brasília: Ms, 2009. 36 p. Disponível em: <http://portal.saude.gov.br/portal/arquivos/pdf/cartilha_acs_dengue_web_09_11.pdf>. Acesso em: 15 jun. 2012.

[6] WROBLEWSKI, Luke. Why Mobile Matters. Disponível em: <http://www.lukew.com/ff/entry.asp?1506>. Acesso em: 10 out. 2012.

[7] NIELSEN (Estados Unidos). More US Consumers Choosing Smartphones as Apple Closes the Gap on Android. Disponível em: <http://blog.nielsen.com/nielsenwire/consumer/more-us-consumers-choosing-smartphones-as-apple-closes-the-gap-on-android/>. Acesso em: 11 out. 2012.

[8] ABLESON, Frank. Introdução ao Desenvolvimento do Android. Site Oficial da IBM. Disponível em:<http://www.ibm.com/developerworks/br/library/os-android-devel/>. Acesso em 13 abr. 2012.

[9] MORIMOTO, Carlos E. Entendendo o Google Android. Revista GdHn (Guia do Hardware), ed. 11, maio, 2008. p. 12 – 18.

[10] Site oficial OHA (Open Handset Alliance). Mantido pela Google que é membro dessa organização. Disponível em: < http://www.openhandsetalliance.com/ >. Acesso em: 14 junho 2012.

[11] PEREIRA, Lucio Camilo; SILVA, Michel. Android para Desenvolvedores, 1. ed. Rio de Janeiro: Editora Brasport, 2009.

[12] FARIA, Alessandro de Oliveira. Programe seu Andróide. Linux Magazine, Volume 1, Número 43, p,73-77,2008.

[13] CATAPAN, Danilo Rodrigues.Desenvolvimento de Aplicativo para Android SDK. 2009. 50 f. Monografia (Graduação) - Curso de Bacharelado em Ciência da Computação, Fema, Assis, 2009.

ANEXO

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Rect;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.OnTouchListener;

public class Bob implements OnTouchListener{

	int x, y, width, height;
	Bitmap b;
	MyView myV;				//atributos de Bob
	int anda = 0, currentFrame = 0, direction = 0;
	static TCCActivity xtela, ytela;
	//variaveis para verificar o lado pressionado
	boolean goUp=false, goDown=false, goLeft=false, goRight=false, colidiu=false;
	//direction caso haja mais de uma linha no sprite sheet
	//currentFrame caso haja mais de uma coluna no sprite sheet

	//construtor
	public Bob(MyView myView, Bitmap bob, TCCActivity xtelav, TCCActivity ytelav) {
		xtela = xtelav;
		ytela = ytelav;
		b = bob;
		myV = myView;
		myV.setOnTouchListener(this);
		height = bob.getHeight() / 4;   //define o tamanho y da sprite
		width = bob.getWidth() / 4;     //define o tamanho x da sprite
		x = 240-16;
		y = 160-16;

	}

    public void colide(boolean valor){
	  colidiu=valor;
	  if(colidiu) {
		if(goUp)ytela.ytela-=3;
		if(goDown)ytela.ytela+=3;
		if(goLeft)xtela.xtela-=3;
		if(goRight)xtela.xtela+=3;
	  }
    }
    public void voltar(){
	  colidiu = false;
    }
```

```
	private void upDate() {
		if(!colidiu){
		if(goDown){
			if(y+b.getHeight() < myV.getHeight()){	//verifica se não bateu no
canto da tela
				ytela.ytela-=2;
	//movimenta tela para baixo
				direction = 0;
				anda++;
				if(anda == 8){
					currentFrame = ++currentFrame % 4;
					anda = 0;
				}
				goUp = goRight = goLeft = false;
				}
		}
		if(goUp){
			if(y > 0){
	//verifica se não bateu no canto da tela
				ytela.ytela+=2;
	//movimenta tela para cima
				direction = 3;
				anda++;
				if(anda == 8){
					currentFrame = ++currentFrame % 4;
					anda = 0;
				}
				goLeft = goRight = goDown = false;
			}
		}
		if(goLeft){
			if(x > 0){
	//verifica se não bateu no canto da tela
				xtela.xtela+=2;
	//movimenta tela para esquerda
				direction = 1;
				anda++;
				if(anda == 8){
					currentFrame = ++currentFrame % 4;
					anda = 0;
				}
				goDown = goRight = goUp = false;
			}
		}
		if(goRight){
			if(x+b.getWidth()/4 < myV.getWidth()){	//verifica se não bateu no
canto da tela
				xtela.xtela-=2;
	//movimenta tela para direita
				direction = 2;
				anda++;
				if(anda == 8){
					currentFrame = ++currentFrame % 4;
					anda = 0;
					}
				goDown = goLeft = goUp = false;
				}
			}
		}
	}
```

```
public void onDraw(Canvas canvas) {
		upDate();
		//controle dos frames da sprite
		int srcX = currentFrame * width;
		int srcY = direction * height;

		Rect src = new Rect(srcX, srcY, srcX + width , srcY + height);
		Rect dst = new Rect(x, y, x+width, y+height);

		canvas.drawBitmap(b, src, dst, null);		//desenha na tela

		Rect ret_down = new Rect();
		Rect ret_right = new Rect();
		Rect ret_left = new Rect();
		Rect ret_up = new Rect();

		ret_down.set (50, 250, 100, 300);
		ret_right.set(100, 200, 150, 250);
		ret_left.set(0, 200, 50, 250);
		ret_up.set(50, 150, 100, 200);

		Paint white = new Paint();
		Paint black = new Paint();

		white.setColor(Color.WHITE);
		white.setStyle(Paint.Style.STROKE);
		black.setStyle(Paint.Style.FILL);

		if(!goDown)canvas.drawRect(ret_down, white);
		else canvas.drawRect(ret_down, black);

		if(!goRight)canvas.drawRect(ret_right, white);
		else canvas.drawRect(ret_right, black);

		if(!goLeft)canvas.drawRect(ret_left, white);
		else canvas.drawRect(ret_left, black);

		if(!goUp)canvas.drawRect(ret_up, white);
		else canvas.drawRect(ret_up, black);
	}
	@Override
	public boolean onTouch(View view, MotionEvent event) {
		//verifica se e onde o usuário tocou na tela
		if(event.getX() >=0 && event.getX() < 50 && event.getY() >= 200 && event.getY() <
250){
			goLeft=true;
		}
		if(event.getX() >=50 && event.getX() < 100 && event.getY() >= 250 && event.getY()
< 300){
			goDown=true;
		}
		if(event.getX() >=100 && event.getX() < 150 && event.getY() >= 200 &&
event.getY() < 250){
			goRight=true;
		}
		if(event.getX() >=50 && event.getX() < 100 && event.getY() >= 150 && event.getY()
< 200){
			goUp=true;
		}
		//assim que o usuário não estiver mais tocando na tela todos recebem false
		if(event.getAction() == MotionEvent.ACTION_UP) goDown = goLeft = goRight = goUp =
false;
		return true;
	}
}
```

```
package tcc.game.dengue;

import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Rect;
import tcc.game.dengue.TCCActivity.MyView;

public class Arvores {
	//propriedades
	int ax, ay, width, height, currentFrame = 4, direction = 0;
	Bitmap a;
	MyView myV;
	//construtor
	public Arvores(MyView myView, Bitmap arv, int x, int y) {
		a = arv;
		myV = myView;
		ax = x;
		ay = y;
		height = a.getHeight()/2;
		width = a.getWidth()/6;
	}
	//método de movimentação
	public void moveArvores(int ptelax, int ptelay){
		ax = ax+ptelax;
		ay = ay+ptelay;
	}
	//método de desenho
	public void onDraw(Canvas canvas) {
		//variáveis utilizadas para controle dos frames da sprite
		int srcX = currentFrame * width;
		int srcY = direction * height;

		Rect src = new Rect(srcX, srcY, srcX + width , srcY+height);
		Rect dst = new Rect(ax, ay, ax+width, ay+height);

		canvas.drawBitmap(a, src, dst, null);
	}
	//calculo do valor absoluto
	private int abs(int valor){
		if(valor<0)
			return (valor*-1);
		return valor;
	}
	//====================================testes de
colisão=======================================//
	public boolean colidexi(int xi, int yi, int xf, int yf, int xarvorei, int yarvorei, int
xarovref, int yarvoref){
		if(abs(xarovref-xi) < 6){
			if((yi >= yarvorei) && (yi <= yarvoref))
				return true;
			if((yf >= yarvorei) && (yf <= yarvoref))
				return true;
			if((yi < yarvorei) && (yf > yarvoref))
				return true;
			if((yarvorei < yi) && (yarvoref > yf))
				return true;

		}
		return false;
	}
```

```
public boolean colideyi(int xi, int yi, int xf, int yf, int xarvorei, int yarovrei, int
xarvoref, int yarvoref){
		if(abs(yarvoref-yi) < 6){
			if((xi >= xarvorei) && (xi <= xarvoref))
				return true;
			if((xf >= xarvorei) && (xf <= xarvoref))
				return true;

		}
		return false;
	}
	public boolean colidexf(int xi, int yi, int xf, int yf, int xarvorei, int yarvorei, int
xarvoref, int yarvoref){
		if(abs(xarvorei-xf) < 6){
			if((yi >= yarvorei) && (yi <= yarvoref))
				return true;
			if((yf >= yarvorei) && (yf <= yarvoref))
				return true;
			if((yi < yarvorei) && (yf > yarvoref))
				return true;
			if((yarvorei < yi) && (yarvoref > yf))
				return true;

		}
		return false;
	}

	public boolean colideyf(int xi, int yi, int xf, int yf, int xarvorei, int yarvorei, int
xarvoref, int yarvoref){
		if(abs(yarvorei-yf) < 6){
			if((xi >= xarvorei) && (xi <= xarvoref))
				return true;
			if((xf >= xarvorei) && (xf <= xarvoref))
				return true;

		}
		return false;
	}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;

public class Caixa{
	//propriedades
	int cx, cy, cwidth, cheight,mqx=1,mqy=1;
	Bitmap c;
	MyView myV;
	//contrutor
	public Caixa(MyView myView, Bitmap caixa, int x, int y) {
		c = caixa;
		myV = myView;
		cx = x;
		cy = y;
		cwidth = c.getWidth();
		cheight = c.getHeight();
	}
	//construtor
	public Caixa(MyView myView, Bitmap caixa, int x, int y, int qx, int qy) {
		c = caixa;
		myV = myView;
		cx = x;
		cy = y;
		cwidth = c.getWidth()*qx;
		cheight = c.getHeight()*qy;
		mqx=qx;
		mqy=qy;
	}
	//método de movimentação
	public void moveCaixa(int ptelax, int ptelay){
		cx = cx+ptelax;
		cy = cy+ptelay;

	}
	//método de desenho este é um pouco mais simples por possuir apenas um frame a ser
imprimido
	public void onDraw(Canvas canvas) {
		for(int i=0;i<mqx;i++){
			for(int j=0;j<mqy;j++){
			   canvas.drawBitmap(c, (i*32)+cx, (j*32)+cy, null);
			}
		}
	}
	//calculo do valor absoluto
	private int abs(int valor){
		if(valor<0)
			return (valor*-1);
		return valor;
	}
	//====================================testes de
colisão=======================================//
	public boolean colidexi(int xi, int yi, int xf, int yf, int xcaixai, int ycaixai, int
xcaixaf, int ycaixaf){
		if(abs(xcaixaf-xi) < 3){
			if((yi > ycaixai) && (yi < ycaixaf))
				return true;
			if((yf > ycaixai) && (yf < ycaixaf))
				return true;

		}
		return false;
	}
```

```
public boolean colideyi(int xi, int yi, int xf, int yf, int xcaixai, int ycaixai, int xcaixaf,
int ycaixaf){
		if(abs(ycaixaf-yi) < 3){
			if((xi >= xcaixai) && (xi < xcaixaf))
				return true;
			if((xf > xcaixai) && (xf < xcaixaf))
				return true;

		}
		return false;
	}
	public boolean colidexf(int xi, int yi, int xf, int yf, int xcaixai, int ycaixai, int
xcaixaf, int ycaixaf){
		if(abs(xcaixai-xf) < 3){
			if((yi > ycaixai) && (yi < ycaixaf))
				return true;
			if((yf > ycaixai) && (yf < ycaixaf))
				return true;

		}
		return false;
	}

	public boolean colideyf(int xi, int yi, int xf, int yf, int xcaixai, int ycaixai, int
xcaixaf, int ycaixaf){
		if(abs(ycaixai-yf) < 3){
			if((xi >= xcaixai) && (xi < xcaixaf))
				return true;
			if((xf > xcaixai) && (xf < xcaixaf))
				return true;

		}
		return false;
	}
	//colisão OO
	//public boolean colide(int x, int y, int width, int height, int xcaixa, int ycaixa, int
cwidth, int cheight){
	//	if(abs(xcaixa-(x+(width))) < 3){
	//		if(y >= ycaixa && y < (ycaixa+cheight))
	//			return true;
	//		if((y+(height)) > ycaixa && (y+(height)) < (ycaixa+cheight))
	//			return true;
	//	}
	//	return false;
	//}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;

public class CaixaCerveja extends Recipientes{
	//atributos
	boolean active = true;
	//construtores herdados da classe Recipientes
	public CaixaCerveja(MyView myView, Bitmap imagem, int x, int y) {
		super(myView, imagem, x, y);
	}
	public CaixaCerveja(MyView myView, Bitmap imagem, int x, int y, int qx, int qy) {
		super(myView, imagem, x, y, qx, qy);
	}
	public void setState(boolean stt){
		active = stt;
	}
	public boolean getState(){
		return active;
	}
	//método de desenho onde só é desenhado se active for true
	public void onDraw(Canvas canvas) {
		if(getState())
		for(int i=0;i<mqx;i++){
			for(int j=0;j<mqy;j++){
			   canvas.drawBitmap(im, (i*32)+rx, (j*32)+ry, null);
			}
		}
	}
}
```

```
package tcc.game.dengue;

import android.graphics.Bitmap;
import android.graphics.Canvas;
import tcc.game.dengue.TCCActivity.MyView;

public class Fundo {
	//atributos
	int x, y, width, height;
	Bitmap f;
	MyView myV;
	//contrutor
	public Fundo(MyView myView, Bitmap fundo) {
		f = fundo;
		myV = myView;
	}
	//método de desenho
	public void onDraw(Canvas canvas) {
		canvas.drawBitmap(f, 0, 0, null);
	}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Rect;

public class CaixaDagua extends Recipientes{
	//atributos
	boolean active = true;
	int currentFrame = 2, direction = 0;
	//contrutores herdados de Recipientes
	public CaixaDagua(MyView myView, Bitmap imagem, int x, int y) {
		super(myView, imagem, x, y);
	}
	public CaixaDagua(MyView myView, Bitmap imagem, int x, int y, int qx, int qy) {
		super(myView, imagem, x, y, qx, qy);
	}
	public void setState(boolean stt){
		active = stt;
	}
	public boolean getState(){
		return active;
	}
	//método de desenho, se active for false a caixa d'agua é tampada
	public void onDraw(Canvas canvas) {
		if(getState()){
			currentFrame = 0;
			for(int i=0;i<mqx;i++){
				for(int j=0;j<mqy;j++){
					int srcX = currentFrame * rwidth/2;
					int srcY = direction * rheight;
					Rect src = new Rect(srcX, srcY, srcX + rwidth/2 ,
srcY+rheight);
					Rect dst = new Rect(rx, ry, rx+rwidth/2, ry+rheight);
					canvas.drawBitmap(im, src, dst, null);
					}
				}
		}
		else {
			currentFrame = 1;
			for(int i=0;i<mqx;i++){
				for(int j=0;j<mqy;j++){
					int srcX = currentFrame * rwidth/2;
					int srcY = direction * rheight;
					Rect src = new Rect(srcX, srcY, srcX + rwidth/2 ,
srcY+rheight);
					Rect dst = new Rect(rx, ry, rx+rwidth/2, ry+rheight);

					canvas.drawBitmap(im, src, dst, null);
					}
				}
		}
	}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Rect;

public class Coracao{
	//atributos
	int x, y, width, height;
	Bitmap b;
	MyView myV;
	int cont = 0, currentFrame = 4, direction = 0;
	TCCActivity xtela, ytela;
	//direction caso haja mais de uma linha no sprite sheet
	//currentFrame caso haja mais de uma coluna no sprite sheet

	public Coracao(MyView myView, Bitmap coracao, int vx, int vy) {
		x = vx;
		y = vy;
		b = coracao;
		myV = myView;
		height = coracao.getHeight();   //define o tamanho y da sprite
		width = coracao.getWidth() / 11; //define o tamanho x da sprite
	}
    //método upDate() é utilizado para modificar o frame da sprite automaticamente
	private void upDate() {
		cont++;
			if(cont == 4){
				currentFrame = ++currentFrame % 11;
				cont = 0;
			}
	}
	public void onDraw(Canvas canvas) {
		upDate();
		int srcX = currentFrame * width;
		int srcY = direction * height;

		Rect src = new Rect(srcX, srcY, srcX + width , srcY + height);
		Rect dst = new Rect(x, y, x+width, y+height);
		canvas.drawBitmap(b, src, dst, null);
	}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;

public class Pet extends Recipientes{
	//atributos
	boolean active = true;
	//contrutores herdados de Recipientes
	public Pet(MyView myView, Bitmap imagem, int x, int y) {
		super(myView, imagem, x, y);
	}
	public Pet(MyView myView, Bitmap imagem, int x, int y, int qx, int qy) {
		super(myView, imagem, x, y, qx, qy);
	}
	public void setState(boolean stt){
		active = stt;
	}
	public boolean getState(){
		return active;
	}
	//método de desenho
	public void onDraw(Canvas canvas) {
		if(getState())
		for(int i=0;i<mqx;i++){
			for(int j=0;j<mqy;j++){
			   canvas.drawBitmap(im, (i*32)+rx, (j*32)+ry, null);
			}
		}

	}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;

public class Pneu extends Recipientes{
	//atributos
	boolean active=true;
	//contrutores herdados de Recipientes
	public Pneu(MyView myView, Bitmap imagem, int x, int y) {
		super(myView, imagem, x, y);
	}
	public Pneu(MyView myView, Bitmap imagem, int x, int y, int qx, int qy) {
		super(myView, imagem, x, y, qx, qy);
	}
	public void setState(boolean stt){
		active = stt;
	}
	public boolean getState(){
		return active;
	}
	//método de desenho
	public void onDraw(Canvas canvas) {
		if(getState())
		for(int i=0;i<mqx;i++){
			for(int j=0;j<mqy;j++){
			   canvas.drawBitmap(im, (i*32)+rx, (j*32)+ry, null);
			}
		}

	}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;

public class Sacola extends Recipientes{
	//atributos
	boolean active = true;
	//contrutores herdados de Recipientes
	public Sacola(MyView myView, Bitmap imagem, int x, int y) {
		super(myView, imagem, x, y);
	}
	public Sacola(MyView myView, Bitmap imagem, int x, int y, int qx, int qy) {
		super(myView, imagem, x, y, qx, qy);
	}
	public void setState(boolean stt){
		active = stt;
	}
	public boolean getState(){
		return active;
	}
	//método de desenho
	public void onDraw(Canvas canvas) {
		if(getState())
		for(int i=0;i<mqx;i++){
			for(int j=0;j<mqy;j++){
			   canvas.drawBitmap(im, (i*32)+rx, (j*32)+ry, null);
			}
		}
	}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;

public class Tampinha extends Recipientes{
	//atributos
	boolean active = true;
	//contrutores herdados da classe Recipientes
	public Tampinha(MyView myView, Bitmap imagem, int x, int y) {
		super(myView, imagem, x, y);
	}
	public Tampinha(MyView myView, Bitmap imagem, int x, int y, int qx, int qy) {
		super(myView, imagem, x, y, qx, qy);
	}
	public void setState(boolean stt){
		active = stt;
	}
	public boolean getState(){
		return active;
	}
	//método de desenho
	public void onDraw(Canvas canvas) {
		if(getState())
		for(int i=0;i<mqx;i++){
			for(int j=0;j<mqy;j++){
			   canvas.drawBitmap(im, (i*32)+rx, (j*32)+ry, null);
			}
		}
	}
}
```

```
package tcc.game.dengue;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;

public class Recipientes {
		//atributos que serão utilizados pelos recipientes
		int rx, ry, rwidth, rheight,mqx=1,mqy=1;
		Bitmap im;
		MyView myV;
		//contrutores que serão utilizados pelos recipientes
		public Recipientes(MyView myView, Bitmap imagem, int x, int y) {
			im = imagem;
			myV = myView;
			rx = x;
			ry = y;
			rwidth = im.getWidth();
			rheight = im.getHeight();
		}
		public Recipientes(MyView myView, Bitmap imagem, int x, int y, int qx, int qy) {
			im = imagem;
			myV = myView;
			rx = x;
			ry = y;
			rwidth = im.getWidth()*qx;
			rheight = im.getHeight()*qy;
			mqx=qx;
			mqy=qy;
		}
//método de movimentação
		public void moveRecipiente(int ptelax, int ptelay){
			rx = rx+ptelax;
			ry = ry+ptelay;
		}
		//método de desenho
		public void onDraw(Canvas canvas) {
			for(int i=0;i<mqx;i++){
				for(int j=0;j<mqy;j++){
				   canvas.drawBitmap(im, (i*32)+rx, (j*32)+ry, null);
				}
			}
		}
		//calculo do valor absoluto para utilizar nas colisões
		private int abs(int valor){
			if(valor<0)
				return (valor*-1);
			return valor;
		}
		//====================================testes de
colisão=======================================//
		public boolean colidexi(int xi, int yi, int xf, int yf, int xrecipientei, int
yrecipientei, int xrecipientef, int yrecipientef){
			if(abs(xrecipientef-xi) < 6){
				if((yi >= yrecipientei) && (yi <= yrecipientef))
					return true;
				if((yf >= yrecipientei) && (yf <= yrecipientef))
					return true;
				if((yi < yrecipientei) && (yf > yrecipientef))
					return true;
				if((yrecipientei < yi) && (yrecipientef > yf))
					return true;
			}
			return false;
		}
```

```
public boolean colideyi(int xi, int yi, int xf, int yf, int xrecipientei, int yrecipientei, int
xrecipientef, int yrecipientef){
                        if(abs(yrecipientef-yi) < 6){
                                if((xi >= xrecipientei) && (xi <= xrecipientef))
                                        return true;
                                if((xf >= xrecipientei) && (xf <= xrecipientef))
                                        return true;
                        }
                        return false;
                }
                public boolean colidexf(int xi, int yi, int xf, int yf, int xrecipientei, int
yrecipientei, int xrecipientef, int yrecipientef){
                        if(abs(xrecipientei-xf) < 6){
                                if((yi >= yrecipientei) && (yi <= yrecipientef))
                                        return true;
                                if((yf >= yrecipientei) && (yf <= yrecipientef))
                                        return true;
                                if((yi < yrecipientei) && (yf > yrecipientef))
                                        return true;
                                if((yrecipientei < yi) && (yrecipientef > yf))
                                        return true;
                        }
                        return false;
                }
                public boolean colideyf(int xi, int yi, int xf, int yf, int xrecipientei, int
yrecipientei, int xrecipientef, int yrecipientef){
                        if(abs(yrecipientei-yf) < 6){
                                if((xi >= xrecipientei) && (xi <= xrecipientef))
                                        return true;
                                if((xf >= xrecipientei) && (xf <= xrecipientef))
                                        return true;
                        }
                        return false;
                }
}
```

```
package tcc.game.dengue;

import java.util.Random;

import tcc.game.dengue.TCCActivity.MyView;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Rect;

public class Pernilongo{

	int perx, pery, perwidth, perheight,mqx=1,mqy=1;
	Bitmap per;
	MyView myV;
	int currentFrame = 3, direction = 0;
	int escolhas[];                                                    //atributos
	boolean estilo = false;
	int xalvo, yalvo, tempoEstilo=400;
	Random randoma;

	public Pernilongo(MyView myView, Bitmap pernilongo, int x, int y) {
		per = pernilongo;
		myV = myView;
		perx = x;
		pery = y;
		perwidth = per.getWidth()/4;
		perheight = per.getHeight();
		randoma = new Random();
		if(randoma.nextInt(2)==0)estilo=false;
		else estilo=true;

	}

	public Pernilongo(MyView myView, Bitmap pernilongo, int x, int y, int qx, int qy) {
		per = pernilongo;
		myV = myView;
		perx = x;
		pery = y;
		perwidth = per.getWidth()*qx;
		perheight = per.getHeight()*qy;
		mqx=qx;
		mqy=qy;
	}



	public void aproximar(int xalvo, int yalvo, int xper, int yper){
		escolhas = new int [4];
		tempoEstilo--;
		if(tempoEstilo<1){
			if(randoma.nextInt(2)==0)estilo=false;
			else estilo=true;
			tempoEstilo=400;
		}

		if(estilo){
			xalvo=this.xalvo;
			yalvo=this.yalvo;
			if(perx==xalvo && pery==yalvo){
				this.xalvo=randoma.nextInt(720);
				this.yalvo=randoma.nextInt(480);
			}
		}
		escolhas[0] = minimaDistancia(xalvo, yalvo, xper, yper-1);
		escolhas[1] = minimaDistancia(xalvo, yalvo, xper, yper+1);
		escolhas[2] = minimaDistancia(xalvo, yalvo, xper-1, yper);
		escolhas[3] = minimaDistancia(xalvo, yalvo, xper+1, yper);
		int menorValor = escolhas[0], indice = 0;
```

```
for(int i = 1; i < 4; i++){
                        if(escolhas[i]<escolhas[i-1]){
                                menorValor = escolhas[i];
                                indice = i;
                        }
                }
                if(indice == 0){
                        pery--;
                        //perx=perx+um*-1;
                        currentFrame=0;
                }
                if(indice == 1){
                        pery++;
                        currentFrame=2;
                }
                if(indice == 2){
                        perx--;
                        currentFrame=1;
                }
                if(indice == 3){
                        perx++;
                        currentFrame=3;
                }

        }

        public int minimaDistancia(int xalvo, int yalvo, int xper, int yper){
                return (abs(xalvo-xper)+abs(yalvo-yper));
        }

        public void movePernilongo(int ptelax, int ptelay){
                perx = perx+ptelax;
                pery = pery+ptelay;

        }
        public void onDraw(Canvas canvas) {
                //canvas.drawBitmap(c, cx, cy, null);

                int srcX = currentFrame * perwidth;
                int srcY = direction * perheight;

                Rect src = new Rect(srcX, srcY, srcX + perwidth , srcY+perheight);
                Rect dst = new Rect(perx, pery, perx+perwidth, pery+perheight);
                canvas.drawBitmap(per, src, dst, null);
        }

        private int abs(int valor){
                if(valor<0)
                        return (valor*-1);
                return valor;
        }

        public boolean colidexi(int xi, int yi, int xf, int yf, int xpernilongoi, int
ypernilongoi, int xpernilongof, int ypernilongof){
                if(abs(xpernilongof-xi) < 6){
                        if((yi >= ypernilongoi) && (yi <= ypernilongof))
                                return true;
                        if((yf >= ypernilongoi) && (yf <= ypernilongof))
                                return true;
                        if((yi < ypernilongoi) && (yf > ypernilongof))
                                return true;
                        if((ypernilongoi < yi) && (ypernilongof > yf))
                                return true;

                }
                return false;
        }
```

```
public boolean colideyi(int xi, int yi, int xf, int yf, int xpernilongoi, int ypernilongoi, int
xpernilongof, int ypernilongof){
		if(abs(ypernilongof-yi) < 6){
			if((xi >= xpernilongoi) && (xi <= xpernilongof))
				return true;
			if((xf >= xpernilongoi) && (xf <= xpernilongof))
				return true;

		}
		return false;
	}
	public boolean colidexf(int xi, int yi, int xf, int yf, int xpernilongoi, int
ypernilongoi, int xpernilongof, int ypernilongof){
		if(abs(xpernilongoi-xf) < 6){
			if((yi >= ypernilongoi) && (yi <= ypernilongof))
				return true;
			if((yf >= ypernilongoi) && (yf <= ypernilongof))
				return true;
			if((yi < ypernilongoi) && (yf > ypernilongof))
				return true;
			if((ypernilongoi < yi) && (ypernilongof > yf))
				return true;

		}
		return false;
	}

	public boolean colideyf(int xi, int yi, int xf, int yf, int xpernilongoi, int
ypernilongoi, int xpernilongof, int ypernilongof){
		if(abs(ypernilongoi-yf) < 6){
			if((xi >= xpernilongoi) && (xi <= xpernilongof))
				return true;
			if((xf >= xpernilongoi) && (xf <= xpernilongof))
				return true;

		}
		return false;
	}
}
```

```
package tcc.game.dengue;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.os.Bundle;
import android.view.MotionEvent;
import android.view.SurfaceHolder;
import android.view.SurfaceView;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.view.View.OnTouchListener;;

public class MenuActivity extends Activity implements OnTouchListener{
	//atributos
	TCCActivity pff;
	MenuView mv;
	Bitmap bm;
	//basicamente o mesmo conceito de TCCActivity o que muda é que nesta tela temos o Intent
	//que é a Intenção de que algo seja feito, neste caso é a chamda da primeira fase
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		requestWindowFeature(Window.FEATURE_NO_TITLE); // sem titulo
	 setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE); // deixa a tela
sempre de lado
	 getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);
		mv = new MenuView(this);
		mv.setOnTouchListener(this);
		bm = BitmapFactory.decodeResource(getResources(), R.drawable.menu);
		setContentView(mv);
	}
	protected void onPause(){
		super.onPause();
		mv.pause();
	}
	protected void onResume() {
		super.onResume();
		mv.resume();
	}
	public class MenuView extends SurfaceView implements Runnable{
		Thread t = null;
		SurfaceHolder holder;
		boolean play = false;

		public MenuView(Context context) {
			super(context);
			holder = getHolder();
		}
		@Override
		protected void onDraw(Canvas canvas) {
			canvas.drawBitmap(bm, 0, 0, null);
			super.onDraw(canvas);
		}
		public void run() {
			while(play){
				if(!holder.getSurface().isValid()) continue;
				Canvas c = holder.lockCanvas();
				onDraw(c);
				holder.unlockCanvasAndPost(c);
			}
		}
```

```
public void pause(){
                    play = false;
                    while(true){
                           try {
                                  t.join();
                           } catch (InterruptedException e) {
                                  e.printStackTrace();
                           }
                           break;
                    }
                    t = null;
             }
             public void resume(){
                    play = true;
                    t = new Thread(this);
                    t.start();
             }
      }
      //ao tocar na tela é chamada outra Activity no caso a primeira fase do jogo
      public boolean onTouch(View v, MotionEvent event) {
             //responsavel pela chamada da primeira fase
             Intent iTCCActivity = new Intent(this, TCCActivity.class);
             startActivity(iTCCActivity);
             return true;
      }
}
```

```
package tcc.game.dengue;

import java.io.IOException;

import android.app.Activity;
import android.content.Context;
import android.content.pm.ActivityInfo;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Paint.Style;
import android.graphics.Typeface;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.view.SurfaceHolder;
import android.view.SurfaceView;
import android.view.Window;
import android.view.WindowManager;

public class TCCActivity extends Activity {
    // declaração de variáveis e objetos que serão utilizados no jogo
       MyView v;                                                                    //visao
    final int LL = 30,CC = 30;                                          //linhas e colunas na
matriz mapa
    Coracao ocoracao, ocoracao2, ocoracao3;                     //vidas
    Fundo ofnd;                                                                  //imagem
de fundo
    Bob obob;                                                                  //agente de
endemias
    Pernilongo opernilongo;               //mosquito da dengue

    Pneu[] vpneu = new Pneu[10];                                //
    Tampinha[] vtampinha = new Tampinha[10];                    //
    Sacola[] vsacola = new Sacola[10];                          //vetores de recipientes
```

```
Pet[] vpet = new Pet[10];                                                   //
    CaixaDagua[] vcaixaDagua = new CaixaDagua[10];       //
    CaixaCerveja[] vcaixaCerveja = new CaixaCerveja[10];//

    Arvores[] varvores = new Arvores[100]; //
    Caixa[] vcaixa = new Caixa[300];              //vetores de obstáculos

    Bitmap bfnd, bmosquito, bcaixa, barvores, bbob, bcoracao, bpneu, btampinha, //imagens
    bsacola, bpet, bcaixaDagua, bcaixaCerveja;
           //dos objetos

    MediaPlayer sfundo;                                           //som

    long startTempo = System.currentTimeMillis(), timer = 30000;                     //tempo
limite e inicial

    int qtdeCaixa = 0, qtdeArvore = 0, qtdePneu = 0, qtdeTampinha = 0, qtdeSacola = 0, qtdePet
= 0,   //variaveis de quantidade
    qtdeCaixaDagua = 0, qtdeCaixaCerveja = 0, qtdeVidas = 3, qtdeRecipientes = 0;
                     //de recipientes e vidas

    int xtela=0, ytela=0, pontuacao = 0;   //variveis utilizadas para movimentação da tela e
pontuação

    boolean retirarRecipiente=false;       //variaveis para teste



/*=========================================================================================*/
                                                  matriz mapa onde
2 = arvore // 3 = pneu // 4 = tampinha // 5 = sacola // 6 = pet // 7 = caixadagua // 8 =
caixacerveja
        >1000 = linha de caixas na horizontal e >2000 = linha de caixas na vertical
*/==========================================================================================*/

    int mapa[][]=//30x30
       {{2030,1028,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2030},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,0,0,2,2008,7,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,1003,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,2,0,0,0,0,5,0},
              {0,0,0,0,0,2016,1008,0,0,0,0,0,0,0,2008,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,2004,2004,6,6,6,0,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,2,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0},
              {0,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,2003,2003,0,0,0,0,0,0,1010,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,2006,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,8,0,0,0,7,0,0,0,0,0,0,0,0,0,0,0,0,7,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,1006,0,0,0,0,0,0,0,0,0,0,0,0,1008,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,1006,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,2,2,6,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,6,0,0,0,0},
              {0,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,2,2,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,2,2,0,0,0,0},
              {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0},
              {0,5,0,0,0,0,0,0,0,5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0},
              {0,1028,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}};
```

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE); // sem titulo
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE); // deixa a tela
sempre de lado
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);
        //linha acima retira o manager deixando o app fullscreen

        v = new MyView(this);       //passando o contexto para o objeto da clase MyView
        v.xtelav=this;
        v.ytelav=this;                      //definindo x e y de v para movimentacao da tela

        //atribuindo os bitmaps as imagens do diretorio
        bfnd = BitmapFactory.decodeResource(getResources(), R.drawable.fundo);
        bmosquito = BitmapFactory.decodeResource(getResources(), R.drawable.mosquito);
        bcaixa = BitmapFactory.decodeResource(getResources(), R.drawable.caixa);
        barvores = BitmapFactory.decodeResource(getResources(), R.drawable.arvorestp);
        bbob = BitmapFactory.decodeResource(getResources(), R.drawable.bob);
        bcoracao = BitmapFactory.decodeResource(getResources(), R.drawable.coracao);
        bpneu = BitmapFactory.decodeResource(getResources(), R.drawable.pneu);
        btampinha = BitmapFactory.decodeResource(getResources(), R.drawable.tampinha);
        bsacola = BitmapFactory.decodeResource(getResources(), R.drawable.sacola);
        bpet = BitmapFactory.decodeResource(getResources(), R.drawable.pet);
        bcaixaDagua = BitmapFactory.decodeResource(getResources(), R.drawable.caixa_dagua);
        bcaixaCerveja = BitmapFactory.decodeResource(getResources(), R.drawable.caixa_cerveja);

        //atribuindo o som ao mp3 do diretorio
        sfundo = MediaPlayer.create(getApplicationContext(), R.drawable.musica);

        //chama a visão v
        setContentView(v);
    }

    //metodo de pausa
       protected void onPause(){
              super.onPause();
              if (sfundo.isPlaying()) sfundo.stop();
              try {
                     sfundo.prepare();
              } catch (IllegalStateException e) {
                     e.printStackTrace();
              } catch (IOException e) {
                     e.printStackTrace();
              }
              v.pause();
       }

       //metodo de retomar
       protected void onResume() {
              super.onResume();
              sfundo.setLooping(true);
              if (!sfundo.isPlaying()){
                     sfundo.start();
              }
              v.resume();
       }
```

```
//classe responsavel pelo controle da tela, criação dos objetos, testes e execução do jogo
	public class MyView extends SurfaceView implements Runnable{

		Thread t = null;
		TCCActivity xtelav, ytelav;
		SurfaceHolder holder;
		boolean play = false;

		public MyView(Context context) {

			super(context);
			holder = getHolder();

		}

		//metodo de execução
		public void run() {
			//variaveis para contagem dos recipientes
			int contCaixa = 0, contArvore = 0, contPneu = 0, contTampinha = 0;
			int contSacola = 0, contPet = 0, contCaixaDagua = 0, contCaixaCerveja = 0;

			//criação dos objetos presentes no jogo
			ofnd = new Fundo(this,bfnd);
			obob = new Bob(this, bbob, xtelav, ytelav);
			opernilongo = new Pernilongo(this, bmosquito, 10, 10);;

			if(qtdeVidas==3){
				ocoracao = new Coracao(this, bcoracao, 0, 0);
				ocoracao2 = new Coracao(this, bcoracao, 50, 0);
				ocoracao3 = new Coracao(this, bcoracao, 100, 0);

			}else if(qtdeVidas==2){
				ocoracao = new Coracao(this, bcoracao, 0, 0);
				ocoracao2 = new Coracao(this, bcoracao, 50, 0);
			}else if(qtdeVidas==1){
				ocoracao = new Coracao(this, bcoracao, 0, 0);
			}
				//caso qtdeVidas seja == 0 o jogo fecha indo pro menu principal
				else finish();
			//criação dos recipientes
			for(int i = 0; i < CC; i++)
				for(int j =0; j<LL;j++){
					if(mapa[i][j] > 1000){
						if((mapa[i][j] - 1000) < 1000){
							vcaixa[contCaixa] = new Caixa(this, bcaixa,
(j*32), (i*32),(mapa[i][j]-1000),1);//32 tamanho da caixa no cel
							contCaixa++;
						}
						if((mapa[i][j] - 1000) > 1000){
							vcaixa[contCaixa] = new Caixa(this, bcaixa,
(j*32), (i*32),1,(mapa[i][j]-2000));//32 tamanho da caixa no cel
							contCaixa++;
						}
					}
					if(mapa[i][j] == 2){
						varvores[contArvore] = new Arvores(this, barvores,
j*32, i*32);//32 tamanho no cel
						contArvore++;
					}
					if(mapa[i][j] == 3){
						vpneu[contPneu] = new Pneu(this, bpneu, j*32,
i*32);//32 tamanho no cel
						contPneu++;
					}
					if(mapa[i][j] == 4){
						vtampinha[contTampinha] = new Tampinha(this,
btampinha, j*32, i*32);//32 tamanho no cel
						contTampinha++;
						}
```

```
if(mapa[i][j] == 5){
                                        vsacola[contSacola] = new Sacola(this, bsacola,
j*32, i*32);//32 tamanho no cel
                                        contSacola++;
                                }
                                if(mapa[i][j] == 6){
                                        vpet[contPet] = new Pet(this, bpet, j*32, i*32);//32
tamanho no cel
                                        contPet++;
                                }
                                if(mapa[i][j] == 7){
                                        vcaixaDagua[contCaixaDagua] = new CaixaDagua(this,
bcaixaDagua, j*32, i*32);//32 tamanho no cel
                                        contCaixaDagua++;
                                }
                                if(mapa[i][j] == 8){
                                        vcaixaCerveja[contCaixaCerveja] = new
CaixaCerveja(this, bcaixaCerveja, j*32, i*32);//32 tamanho no cel
                                        contCaixaCerveja++;
                                }
                        }
                        qtdeCaixa=contCaixa;
                        qtdeArvore=contArvore;
                        qtdePneu=contPneu;
                        qtdeTampinha=contTampinha;
                        qtdeSacola=contSacola;
                        qtdePet=contPet;
                        qtdeCaixaDagua=contCaixaDagua;
                        qtdeCaixaCerveja=contCaixaCerveja;

        qtdeRecipientes=qtdePneu+qtdeTampinha+qtdeSacola+qtdePet+qtdeCaixaDagua+qtdeCaixaCerveja
;
                        //qtdeRecipientes recebe a soma de todos

//=============================================================================================//
//                                    execução do jogo                                         //
//=============================================================================================//
                        while(play){
                                if(!holder.getSurface().isValid()) continue;
                                contCaixa=0;
                                contArvore=0;
                                contPneu=0;
                                contTampinha=0;
                                contSacola=0;
                                contPet=0;
                                contCaixaDagua=0;
                                contCaixaCerveja=0;
                                //movimenta o pernilongo
                                opernilongo.movePernilongo(xtelav.xtela, ytelav.ytela);
                                //calcula a direção que o pernilongo irá movimentar
                                opernilongo.aproximar(obob.x, obob.y, opernilongo.perx,
opernilongo.pery);
                //movimentação de todos os objetos pois quem se movimenta não é o bob
                //ele fica "preso" no centro da tela
                                for(int i = 0; i < CC; i++)
                                        for(int j =0; j < LL;j++){
                                                if(mapa[i][j] > 1000){
                                                        vcaixa[contCaixa].moveCaixa(xtelav.xtela,
ytelav.ytela);
                                                        contCaixa++;
                                                }
                                                else if(mapa[i][j] == 2){

        varvores[contArvore].moveArvores(xtelav.xtela, ytelav.ytela);
                                                        contArvore++;
                                                }
```

```
else if(mapa[i][j] == 3){
                                                vpneu[contPneu].moveRecipiente(xtelav.xtela,
ytelav.ytela);
                                                contPneu++;
                                        }
                                        else if(mapa[i][j] == 4){

       vtampinha[contTampinha].moveRecipiente(xtelav.xtela, ytelav.ytela);
                                                contTampinha++;
                                        }
                                        else if(mapa[i][j] == 5){

       vsacola[contSacola].moveRecipiente(xtelav.xtela, ytelav.ytela);
                                                contSacola++;
                                        }
                                        else if(mapa[i][j] == 6){
                                                vpet[contPet].moveRecipiente(xtelav.xtela,
ytelav.ytela);
                                                contPet++;
                                        }
                                        else if(mapa[i][j] == 7){

       vcaixaDagua[contCaixaDagua].moveRecipiente(xtelav.xtela, ytelav.ytela);
                                                contCaixaDagua++;
                                        }
                                        else if(mapa[i][j] == 8){

       vcaixaCerveja[contCaixaCerveja].moveRecipiente(xtelav.xtela, ytelav.ytela);
                                                contCaixaCerveja++;
                                        }
                                }

                        xtelav.xtela=0;
                        ytelav.ytela=0;
//======================//                testes                //======================//
                //torna o valor da colisão de obob = a false
                        obob.voltar();
//verifica a colisão com as caixas que são as paredes
                        for(int i =0 ; i<qtdeCaixa; i++){
                                obob.colide(vcaixa[i].colidexi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vcaixa[i].cx, vcaixa[i].cy,
vcaixa[i].cwidth+vcaixa[i].cx, vcaixa[i].cheight+vcaixa[i].cy));//esquerda
                                obob.colide(vcaixa[i].colideyi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vcaixa[i].cx, vcaixa[i].cy,
vcaixa[i].cwidth+vcaixa[i].cx, vcaixa[i].cheight+vcaixa[i].cy));//cima
                                obob.colide(vcaixa[i].colidexf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vcaixa[i].cx, vcaixa[i].cy,
vcaixa[i].cwidth+vcaixa[i].cx, vcaixa[i].cheight+vcaixa[i].cy));//direita
                                obob.colide(vcaixa[i].colideyf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vcaixa[i].cx, vcaixa[i].cy,
vcaixa[i].cwidth+vcaixa[i].cx, vcaixa[i].cheight+vcaixa[i].cy));//baixo
                        }
                        //verifica a colisão com as arvores
                        for(int i =0 ; i<qtdeArvore; i++){
                                obob.colide(varvores[i].colidexi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, varvores[i].ax, varvores[i].ay,
varvores[i].width+varvores[i].ax, varvores[i].height+varvores[i].ay));//esquerda
                                obob.colide(varvores[i].colideyi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, varvores[i].ax, varvores[i].ay,
varvores[i].width+varvores[i].ax, varvores[i].height+varvores[i].ay));//cima
                                obob.colide(varvores[i].colidexf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, varvores[i].ax, varvores[i].ay,
varvores[i].width+varvores[i].ax, varvores[i].height+varvores[i].ay));//direita
                                obob.colide(varvores[i].colideyf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, varvores[i].ax, varvores[i].ay,
varvores[i].width+varvores[i].ax, varvores[i].height+varvores[i].ay));//baixo
                        }
```

```
//verifica a colisão com os pneus aumentando a pontuação e diminuindo a qtde de recipientes
                            //setState(false) retira o objeto da tela
                            for(int i =0 ; i<qtdePneu; i++){
                                   if(vpneu[i].getState()){
                                          obob.colide(vpneu[i].colidexi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vpneu[i].rx, vpneu[i].ry, vpneu[i].rwidth+vpneu[i].rx,
vpneu[i].rheight+vpneu[i].ry));//esquerda
                                          if(obob.colidiu){
                                                 vpneu[i].setState(false);
                                                 qtdeRecipientes--;
                                                 pontuacao+=10;
                                          }
                                          obob.colide(vpneu[i].colideyi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vpneu[i].rx, vpneu[i].ry, vpneu[i].rwidth+vpneu[i].rx,
vpneu[i].rheight+vpneu[i].ry));//cima
                                          if(obob.colidiu){
                                                 vpneu[i].setState(false);
                                                 qtdeRecipientes--;
                                                 pontuacao+=10;
                                          }
                                          obob.colide(vpneu[i].colidexf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vpneu[i].rx, vpneu[i].ry, vpneu[i].rwidth+vpneu[i].rx,
vpneu[i].rheight+vpneu[i].ry));//direita
                                          if(obob.colidiu){
                                                 vpneu[i].setState(false);
                                                 qtdeRecipientes--;
                                                 pontuacao+=10;
                                          }
                                          obob.colide(vpneu[i].colideyf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vpneu[i].rx, vpneu[i].ry, vpneu[i].rwidth+vpneu[i].rx,
vpneu[i].rheight+vpneu[i].ry));//baixo
                                          if(obob.colidiu){
                                                 vpneu[i].setState(false);
                                                 qtdeRecipientes--;
                                                 pontuacao+=10;
                                          }      }      }
                            //verifica a colisão com as tampinhas aumentando a pontuação e
diminuindo a qtde de recipientes
                            //setState(false) retira o objeto da tela
                            for(int i =0 ; i<qtdeTampinha; i++){
                                   if(vtampinha[i].getState()){
                                          obob.colide(vtampinha[i].colidexi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vtampinha[i].rx, vtampinha[i].ry,
vtampinha[i].rwidth+vtampinha[i].rx, vtampinha[i].rheight+vtampinha[i].ry));//esquerda
                                          if(obob.colidiu){
                                                 vtampinha[i].setState(false);
                                                 qtdeRecipientes--;
                                                 pontuacao+=10;
                                          }
                                          obob.colide(vtampinha[i].colideyi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vtampinha[i].rx, vtampinha[i].ry,
vtampinha[i].rwidth+vtampinha[i].rx, vtampinha[i].rheight+vtampinha[i].ry));//cima
                                          if(obob.colidiu){
                                                 vtampinha[i].setState(false);
                                                 qtdeRecipientes--;
                                                 pontuacao+=10;
                                          }
                                          obob.colide(vtampinha[i].colidexf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vtampinha[i].rx, vtampinha[i].ry,
vtampinha[i].rwidth+vtampinha[i].rx, vtampinha[i].rheight+vtampinha[i].ry));//direita
                                          if(obob.colidiu){
                                                 vtampinha[i].setState(false);
                                                 qtdeRecipientes--;
                                                 pontuacao+=10;
                                          }
```

```
obob.colide(vtampinha[i].colideyf(obob.x, obob.y, obob.x+obob.width, obob.y+obob.height,
vtampinha[i].rx, vtampinha[i].ry, vtampinha[i].rwidth+vtampinha[i].rx,
vtampinha[i].rheight+vtampinha[i].ry));//baixo
                                        if(obob.colidiu){
                                                vtampinha[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
                                }
                        }
                        //verifica a colisão com os sacola aumentando a pontuação e
diminuindo a qtde de recipientes
                        //setState(false) retira o objeto da tela
                        for(int i =0 ; i<qtdeSacola; i++){
                                if(vsacola[i].getState()){
                                        obob.colide(vsacola[i].colidexi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vsacola[i].rx, vsacola[i].ry,
vsacola[i].rwidth+vsacola[i].rx, vsacola[i].rheight+vsacola[i].ry));//esquerda
                                        if(obob.colidiu){
                                                vsacola[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
                                        obob.colide(vsacola[i].colideyi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vsacola[i].rx, vsacola[i].ry,
vsacola[i].rwidth+vsacola[i].rx, vsacola[i].rheight+vsacola[i].ry));//cima
                                        if(obob.colidiu){
                                                vsacola[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
                                        obob.colide(vsacola[i].colidexf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vsacola[i].rx, vsacola[i].ry,
vsacola[i].rwidth+vsacola[i].rx, vsacola[i].rheight+vsacola[i].ry));//direita
                                        if(obob.colidiu){
                                                vsacola[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
                                        obob.colide(vsacola[i].colideyf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vsacola[i].rx, vsacola[i].ry,
vsacola[i].rwidth+vsacola[i].rx, vsacola[i].rheight+vsacola[i].ry));//baixo
                                        if(obob.colidiu){
                                                vsacola[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
                                }
                        }
                        //verifica a colisão com as pets aumentando a pontuação e
diminuindo a qtde de recipientes
                        //setState(false) retira o objeto da tela
                        for(int i =0 ; i<qtdePet; i++){
                                if(vpet[i].getState()){
                                        obob.colide(vpet[i].colidexi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vpet[i].rx, vpet[i].ry, vpet[i].rwidth+vpet[i].rx,
vpet[i].rheight+vpet[i].ry));//esquerda
                                        if(obob.colidiu){
                                                vpet[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
```

```
obob.colide(vpet[i].colideyi(obob.x, obob.y, obob.x+obob.width, obob.y+obob.height, vpet[i].rx,
vpet[i].ry, vpet[i].rwidth+vpet[i].rx, vpet[i].rheight+vpet[i].ry));//cima
                                        if(obob.colidiu){
                                                vpet[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
                                        obob.colide(vpet[i].colidexf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vpet[i].rx, vpet[i].ry, vpet[i].rwidth+vpet[i].rx,
vpet[i].rheight+vpet[i].ry));//direita
                                        if(obob.colidiu){
                                                vpet[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
                                        obob.colide(vpet[i].colideyf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vpet[i].rx, vpet[i].ry, vpet[i].rwidth+vpet[i].rx,
vpet[i].rheight+vpet[i].ry));//baixo
                                        if(obob.colidiu){
                                                vpet[i].setState(false);
                                                qtdeRecipientes--;
                                                pontuacao+=10;
                                        }
                                }
                        }
                        //verifica a colisão com as caixas d'agua aumentando a pontuação e
diminuindo a qtde de recipientes
                        //setState(false) tampa a caixa d'agua
                        for(int i =0 ; i<qtdeCaixaDagua; i++){
                                obob.colide(vcaixaDagua[i].colidexi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vcaixaDagua[i].rx, vcaixaDagua[i].ry,
vcaixaDagua[i].rwidth/2+vcaixaDagua[i].rx,
vcaixaDagua[i].rheight+vcaixaDagua[i].ry));//esquerda
                                if(obob.colidiu && vcaixaDagua[i].getState()){
                                        vcaixaDagua[i].setState(false);
                                        qtdeRecipientes--;
                                        pontuacao+=10;
                                }
                                obob.colide(vcaixaDagua[i].colideyi(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vcaixaDagua[i].rx, vcaixaDagua[i].ry,
vcaixaDagua[i].rwidth/2+vcaixaDagua[i].rx, vcaixaDagua[i].rheight+vcaixaDagua[i].ry));//cima
                                if(obob.colidiu && vcaixaDagua[i].getState()){
                                        vcaixaDagua[i].setState(false);
                                        qtdeRecipientes--;
                                        pontuacao+=10;
                                }
                                obob.colide(vcaixaDagua[i].colidexf(obob.x, obob.y,
obob.x+obob.width, obob.y+obob.height, vcaixaDagua[i].rx, vcaixaDagua[i].ry,
vcaixaDagua[i].rwidth/2+vcaixaDagua[i].rx, vcaixaDagua[i].rheight+vcaixaDagua[i].ry));//direita
                                if(obob.colidiu && vcaixaDagua[i].getState()){
                                        vcaixaDagua[i].setState(false);
                                        qtdeRecipientes--;
                                        pontuacao+=10;
                                }
```

```
obob.colide(vcaixaDagua[i].colideyf(obob.x, obob.y, obob.x+obob.width, obob.y+obob.height,
vcaixaDagua[i].rx, vcaixaDagua[i].ry, vcaixaDagua[i].rwidth/2+vcaixaDagua[i].rx,
vcaixaDagua[i].rheight+vcaixaDagua[i].ry));//baixo
                                    if(obob.colidiu && vcaixaDagua[i].getState()){
                                            vcaixaDagua[i].setState(false);
                                            qtdeRecipientes--;
                                            pontuacao+=10;
                                    }
                            }
                            //verifica a colisão com as caixas de cerveja aumentando a
pontuação e diminuindo a qtde de recipientes
                            //setState(false) retira o objeto da tela
                            for(int i =0 ; i<qtdeCaixaCerveja; i++){
                                    if(vcaixaCerveja[i].getState()){
                                            obob.colide(vcaixaCerveja[i].colidexi(obob.x,
obob.y, obob.x+obob.width, obob.y+obob.height, vcaixaCerveja[i].rx, vcaixaCerveja[i].ry,
vcaixaCerveja[i].rwidth+vcaixaCerveja[i].rx,
vcaixaCerveja[i].rheight+vcaixaCerveja[i].ry));//esquerda
                                            if(obob.colidiu){
                                                    vcaixaCerveja[i].setState(false);
                                                    qtdeRecipientes--;
                                                    pontuacao+=10;
                                            }
                                            obob.colide(vcaixaCerveja[i].colideyi(obob.x,
obob.y, obob.x+obob.width, obob.y+obob.height, vcaixaCerveja[i].rx, vcaixaCerveja[i].ry,
vcaixaCerveja[i].rwidth+vcaixaCerveja[i].rx,
vcaixaCerveja[i].rheight+vcaixaCerveja[i].ry));//cima
                                            if(obob.colidiu){
                                                    vcaixaCerveja[i].setState(false);
                                                    qtdeRecipientes--;
                                                    pontuacao+=10;
                                            }
                                            obob.colide(vcaixaCerveja[i].colidexf(obob.x,
obob.y, obob.x+obob.width, obob.y+obob.height, vcaixaCerveja[i].rx, vcaixaCerveja[i].ry,
vcaixaCerveja[i].rwidth+vcaixaCerveja[i].rx,
vcaixaCerveja[i].rheight+vcaixaCerveja[i].ry));//direita
                                            if(obob.colidiu){
                                                    vcaixaCerveja[i].setState(false);
                                                    qtdeRecipientes--;
                                                    pontuacao+=10;
                                            }
                                            obob.colide(vcaixaCerveja[i].colideyf(obob.x,
obob.y, obob.x+obob.width, obob.y+obob.height, vcaixaCerveja[i].rx, vcaixaCerveja[i].ry,
vcaixaCerveja[i].rwidth+vcaixaCerveja[i].rx,
vcaixaCerveja[i].rheight+vcaixaCerveja[i].ry));//baixo
                                            if(obob.colidiu){
                                                    vcaixaCerveja[i].setState(false);
                                                    qtdeRecipientes--;
                                                    pontuacao+=10;
                                            }
                                    }
                            }
                            //teste qtdeRecipientes
                            if(qtdeRecipientes==0){
                                    finish();
                            }
                            //tempo para coletar recipientes
                            //caso chegue ao tempo limite é retirado uma vida
                            if (System.currentTimeMillis() - startTempo > timer) {
                                    qtdeVidas--;
                                    startTempo=System.currentTimeMillis();
                            }
```

```
//teste de colisão com o pernilongo
                            if((opernilongo.colidexi(obob.x, obob.y, obob.x+obob.width,
obob.y+obob.height, opernilongo.perx, opernilongo.pery, opernilongo.perx+opernilongo.perwidth,
opernilongo.pery+opernilongo.perheight)) ||
                                (opernilongo.colidexf(obob.x, obob.y, obob.x+obob.width,
obob.y+obob.height, opernilongo.perx, opernilongo.pery, opernilongo.perx+opernilongo.perwidth,
opernilongo.pery+opernilongo.perheight)) ||
                                (opernilongo.colideyi(obob.x, obob.y, obob.x+obob.width,
obob.y+obob.height, opernilongo.perx, opernilongo.pery, opernilongo.perx+opernilongo.perwidth,
opernilongo.pery+opernilongo.perheight)) ||
                                (opernilongo.colideyf(obob.x, obob.y, obob.x+obob.width,
obob.y+obob.height, opernilongo.perx, opernilongo.pery, opernilongo.perx+opernilongo.perwidth,
opernilongo.pery+opernilongo.perheight))){
                                    qtdeVidas--;
                                    opernilongo.perx = 10;
                                    opernilongo.pery = 10;
                            }
                            //responsaveis por desenhar os objetos na tela
                            Canvas c = holder.lockCanvas();
                            onDraw(c);
                            holder.unlockCanvasAndPost(c);
                    }
            }
            //metodo reponsavel por desenhar os objetos
            protected void onDraw(Canvas canvas){
                    //desenha a tela de fundo
                    ofnd.onDraw(canvas);
                    Paint p = new Paint();
                    Paint pRect = new Paint();
                    //muda a fonte utilizado na pontuação e timer
                    Typeface font = Typeface.createFromAsset(getContext().getAssets(),
"fonts/custom.TTF");
                    //define a cor e estilo do retangulo que fica atrás da pontuação
                    pRect.setColor(Color.BLACK);
                    pRect.setStyle(Style.FILL_AND_STROKE);
                    //modifica as propriedades da fonte definida anteriormente
                    p.setColor(Color.WHITE);
                    p.setTextSize(20);
                    p.setTypeface(font);
                    p.setAntiAlias(true);

                    //desenha os objetos na tela
                    for(int i = 0; i <qtdeCaixa; i++)vcaixa[i].onDraw(canvas);
                    for(int i = 0; i <qtdeArvore; i++)varvores[i].onDraw(canvas);
                    for(int i = 0; i <qtdePneu; i++)vpneu[i].onDraw(canvas);
                    for(int i = 0; i <qtdeTampinha; i++)vtampinha[i].onDraw(canvas);
                    for(int i = 0; i <qtdeSacola; i++)vsacola[i].onDraw(canvas);
                    for(int i = 0; i <qtdePet; i++)vpet[i].onDraw(canvas);
                    for(int i = 0; i <qtdeCaixaDagua; i++)vcaixaDagua[i].onDraw(canvas);
                    for(int i = 0; i <qtdeCaixaCerveja; i++)vcaixaCerveja[i].onDraw(canvas);
obob.onDraw(canvas);
                    opernilongo.onDraw(canvas);
                    //retangulo que fica atrá da pontuação
                    canvas.drawRect(0, 0, canvas.getWidth(), 35, pRect);
                    //pontuação
                    canvas.drawText("Pontos: "+pontuacao, 380, 20, p);
                    //timer em segundos
                    canvas.drawText("Tempo: "+ (((System.currentTimeMillis()+timer)-
startTempo)-20000)/1000, 200, 20, p);
```

```
//desenha as vidas seguindo as condições
                        if(qtdeVidas==3){
                                ocoracao.onDraw(canvas);
                                ocoracao2.onDraw(canvas);
                                ocoracao3.onDraw(canvas);
                        }else if(qtdeVidas==2){
                                ocoracao.onDraw(canvas);
                                ocoracao2.onDraw(canvas);
                        }else if(qtdeVidas==1)ocoracao.onDraw(canvas);
                        else {
                                finish();
                        }
                        //obs.: a sequencia dos objetos a serem desenhados na tela devem estar na
ordem certa
                        //senão eles podem ser imprimidos um por cima do outro
                        //ex: se desehar a pontuação primeiro e depois o retangula que deveria
ficar atrás
                        //a pontuação não ficará visível
                }
                //método que pausa
                public void pause(){
                        play = false;
                        while(true){
                                try {
                                        t.join();
                                } catch (InterruptedException e) {
                                        e.printStackTrace();
                                }
                                break;
                        }
                        try {
                                Thread.sleep(50);
                        } catch (InterruptedException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }
                        finish();
                        t = null;
                }
                //método que retoma
                public void resume(){
                        play = true;
                        t = new Thread(this);
                        t.start();
                }
    }
}
```