

Instituto Superior de Engenharia de Coimbra

Departamento de Informática

No âmbito da disciplina de Programação em java para dispositivos moveis

Componente prática

Lecionada por: Ricardo Pereira

brancoRelatório - Minesweeper

Realizado por:

Bruno Sousa nº21260212 TPSI

Tiago Correia nº21260205 TPSI

Índice

[Introdução 3](#_Toc504936262)

[Parte gráfica 4](#_Toc504936263)

[Layout menu do jogo 4](#_Toc504936264)

[Layout dos créditos 4](#_Toc504936265)

[Layout de configuração de jogo 5](#_Toc504936266)

[Layout de jogo 5](#_Toc504936267)

[Idiomas 6](#_Toc504936268)

[Ficheiro Manifest 7](#_Toc504936269)

[Classes 8](#_Toc504936270)

[Block 8](#_Toc504936271)

[Mainmenu 9](#_Toc504936272)

[Créditos 10](#_Toc504936273)

[Gamesetting 10](#_Toc504936274)

[MainActivity 12](#_Toc504936275)

[Conclusão 23](#_Toc504936276)

# Introdução

O presente trabalho é sobre a criação de um jogo para android, minesweeper, onde o jogador terá de abrir todas os blocos, que não contenha bomba, num campo de ‘X’ por ‘X’, tanto o numero de bombas como a dimensão do campo são dadas por o jogador, de forma a ganhar o jogo.

O objetivo deste trabalho é o desenvolvimento do jogo funcional para o sistema android no qual se possa jogar tanto sozinho como a dois jogadores.

Está organizado por 4 capítulos, no primeiro é descrito os layouts criados para a aplicação, no segundo capitulo são descritos os idiomas suportados, no terceiro capitulo trata as alterações feitas no ficheiro de manifesto e por ultimo, no quarto capitulo, é explicado todo o código desenvolvido ao longo da aplicação

# Parte gráfica

# Layout menu do jogo

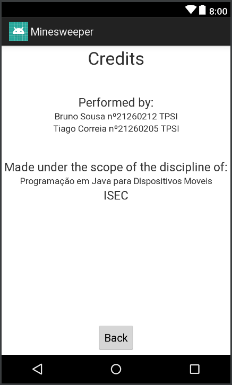
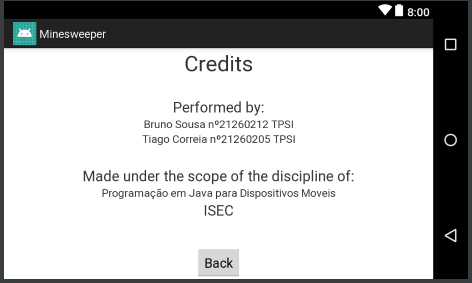
Este layout foi criado para que nos permitisse ter o nome do jogo, com uma textview, e possuir três botões, 1 jogador, 2 jogadores e créditos, todos estes objetos estão dentro de layouts para que se possa centrar no ecrã.

# Layout dos créditos

Este layout apenas contas com varias textview’s no qual metemos as informações a ser mostradas, de modo a que tudo fica-se organizado com queríamos.

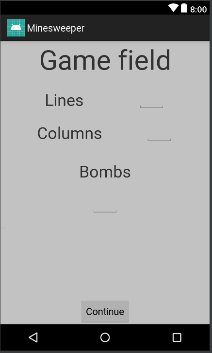
No fim conta com um Butão de voltar para o menu anterior, com a mesma função que o voltar na barra de controlo do android.

# Layout de configuração de jogo

Este layout contem alguns textview’s para mostrar os textos necessários, três edittext para que se meta os valores a utilizar e um botão continuar para irmos para o jogo.

Neste layout foi necessário fazer verificações para se saber se os valor foram ou não introduzidos e se estão dentro dos limites definidos, para que seja possível criar o jogo, caso isto não se verifique o jogo mostra uma mensagem no ecrã a dizer onde esta o problema e não deixa prosseguir ate que todos os campos estejam preenchidos dentro dos parâmetros.

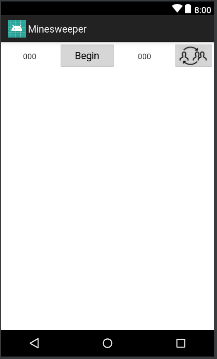
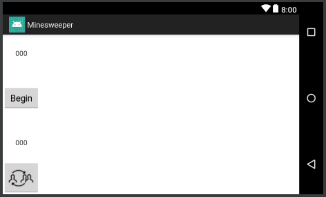
 

# Layout de jogo

Este layout será usado para se jogar, conta com 2 textview’s, uma que faz a contagem do tempo de jogo quando este começa e a outra mostrará o numero de bombas que o utilizador tem no seu campo de jogo.

Temos também 2 botões, um deles serve para iniciar ou recomeçar o jogo, já o outro permite-nos alternar entre o modo de 1 e 2 jogadores.

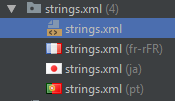
Para mostrar o campo de jogo existe uma linear layout que recebe o campo de jogo tal como o utilizador configurou na atividade anterior.

# Idiomas

A aplicação conta com 4 idiomas que serão auto selecionadas por o sistema, dependendo da linguagem que este tem.

Como default temos o idioma inglês, no entanto caso o sistema do utilizador esteja em português, francês ou japonês, todos os textos serão traduzidos, pois foram criados ficheiros strings.xml para essas.



# Ficheiro Manifest

Neste ficheiro foram adicionadas todas as classes que podiam iniciar uma atividade na aplicação e também configurada de maneira a que o sistema saiba qual o tipo de intente e a sua categoria.

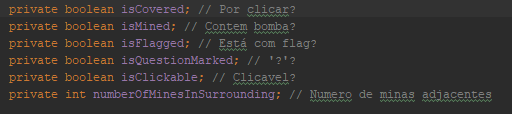


# Classes

# Block

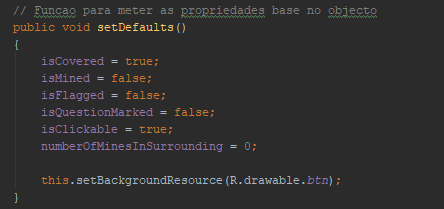
Esta classe será usada para criar do campo de jogo, pois faz ‘extends Button’, para que assim quando criamos um objeto desta classe, tenha por base do botão.

As variáveis criadas na nesta classe são usadas para verificações que precisamos quando estamos a jogar.



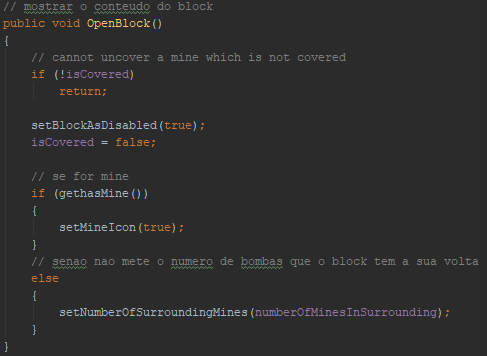
Aparte disto, temos nesta classe o construtor, gets, para que seja possível obter a informação das variáveis pois estas são privadas e não podem ser acedidas diretamente de outra classe, os sets e ainda algumas funções.

A função setDefault faz com que o objeto criado fique com os valores iniciais que são necessários e define também qual o seu aspeto.

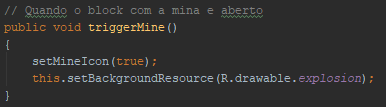


A função Openblock verifica se o bloco esta ou não tapado, caso já não esteja sai da função, no caso de estar ainda, vai meter o bloco desativo e vai mete-lo como não tapado.

No caso de o bloco ter uma bomba, chamamos a função que trata das bombas, senão metemos o aspeto do bloco com o numero de bombas que tem a volta com a função Set para o efeito.



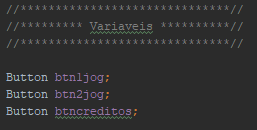
Quando clicamos numa bomba esta terá a imagem alterada para uma explosão.



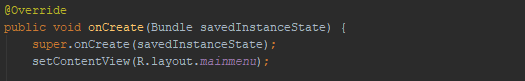
# Mainmenu

Esta classe é a primeira a ser executada pois e o primeiro menu que temos no jogo.

Criamos três variáveis do tipo botões para que sejam associados aos existentes no layout.

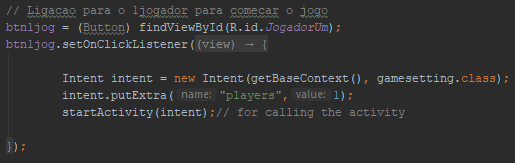


No método onCreate vamos fazer o setContent e dizer ao sistema qual o layout a ser carregado para o utilizador.



Depois disso vamos então ligar as variáveis criadas nesta classe aos botões presentes no layout que carregados, esta ligação é feita por Id do objeto, que é criado no ficheiro xml do layout.

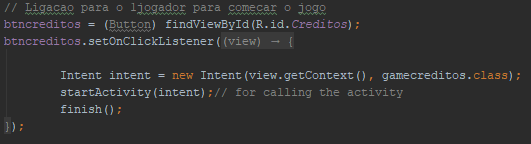
Já com esta ligação, vamos agora fazer a função para que quando se carregue no botão, este crie um intent que será para chamar a classe da configuração de jogo, no entanto como temos 2 modos de jogo, será posto no intent uma variável extra que dirá à próxima atividade qual dos botões foi clicado.



A variável terá o nome de players, do tipo inteiro e o seu conteúdo será ‘1’ no caso de ser clicado o botão para 1 jogador e ‘2’ no 2 jogadores.

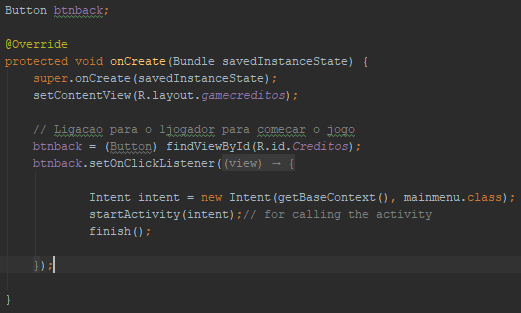


No botão para os créditos não precisamos de nenhuma informação extra, no entanto apos fazer o startActivity, a atividade do menu e fechada, para que quando voltamos da atividade dos créditos não seja criada outra instancia do menu e fiquemos com 2 na pilha de atividades



# Créditos

Nesta atividade apenas temos alguma informação relativa ao jogo e um botão que fazemos a ligação ao criado no layout e que ao clicar irá chamar a atividade do menu principal e fechar a própria atividade.



# Gamesetting

Esta atividade é usada para que o utilizador configure o campo de jogo da maneira que quiser, no entanto existem limites para esses valores.

No onCreate vamos então buscar os dados que enviamos no intent a partir da outra atividade e guarda-los para que se possa enviar depois para a janela de jogo e esta saiba que modo queremos jogar.



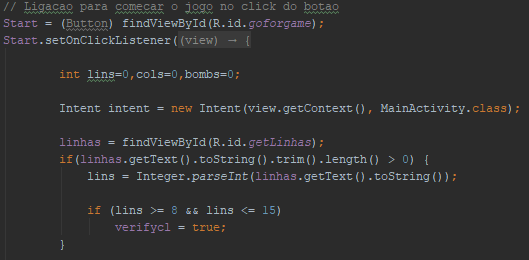
No botão para avançar para a atividade do campo de jogo, vamos ter de fazer verificações e caso estas não se confirmem não se pode avançar para a outra atividade.

Estas verificações são se o utilizador introduziu ou não valores nos edittext destinamos ao numero de colunas, linhas e bombas.

Esta verificação é feita ao tamanho dos dados introduzidos, no caso de ser 0 sabemos que não existem dados em 1 ou mais campos.

Quando os dados introduzidos passam nesta verificação é feita uma segunda para sabermos os dados estão entre os limites definidos.

Para a linhas e colunas os limites são no mínimo 8 e no máximo 15.

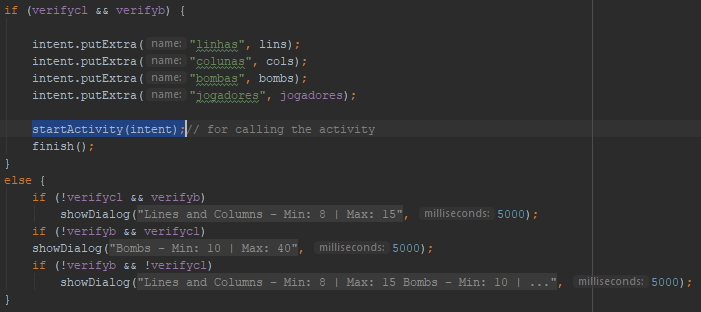


Já para as bombas que podemos ter o limite mínimo é 10 e o máximo 40.



Quando todos os requisitos são passados, guardamos no intent os dados e o modo de jogo que veio da atividade anterior, para que sejam enviados para a nova atividade.

No caso de algo ter falhado, mostramos uma mensagem ao utilizador a dizer onde podem estar os erros para que assim o utilizador saiba o que fazer.

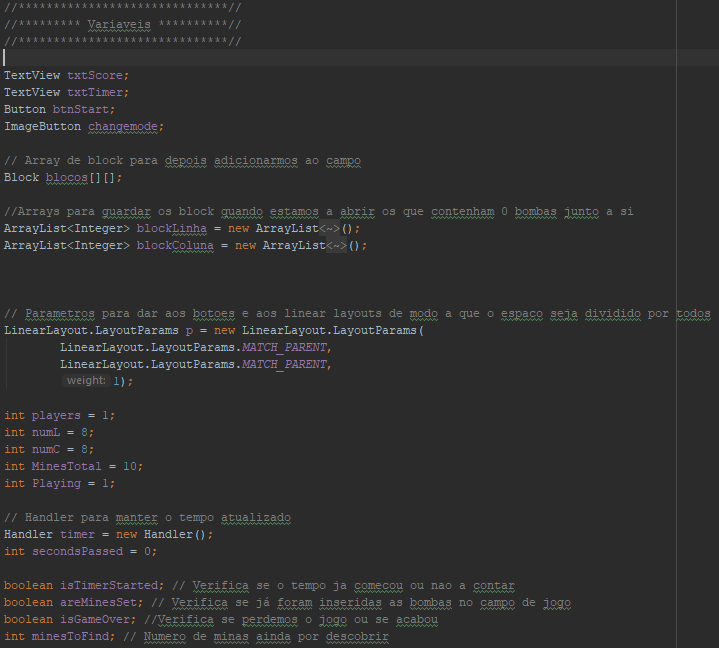


# MainActivity

Nesta classe vamos ter um array de blocos que vai ser usado para fazer o campo de jogo, dois arraylists de inteiros que são usados por a função que abre as casas adjacentes de um bloco sem bombas próximas, um layoutparams que é usado para darmos o mesmo peso a todos os objetos que vamos criar para que fiquem com o peso igual e assim façam a divisão do espaço na view de forma igual.

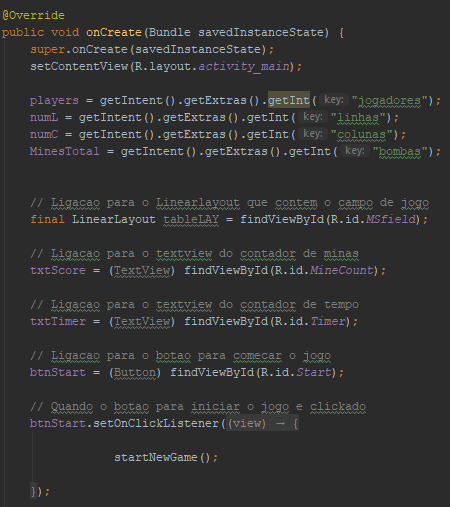
Existem também variáveis para que são usadas para efeitos de jogo, a ‘players’ diz-nos o modo em que estamos, ‘numL’ e ‘numC’ são o numero de colunas e linhas do campo de jogo, ‘minestotal’ são as bombas que serão postas no campo de jogo, ‘Playing’ diz-nos o turno do jogador que irá fazer a jogada, mas apenas no modo de 2 jogadores, ‘istimerstarted’ serve para sabermos se o tempo de jogo já esta a contar ou não, ‘areminesset’ para sabermos se as bombas já foram introduzidas no campo de jogo, isto porque só após a primeira jogada é que são postas as bombas, ‘isgameover’ contem a informação se acabou o jogo ou não e temos a ‘minestofind’ que e usada para mostrar ao utilizador o numero de bombas que estão no campo sem estarem marcadas com bandeira.

Foi também criado um handler para ser usado na contagem do tempo de jogo e uma variável para meter o tempo e depois mostrar o mesmo no ecrã.



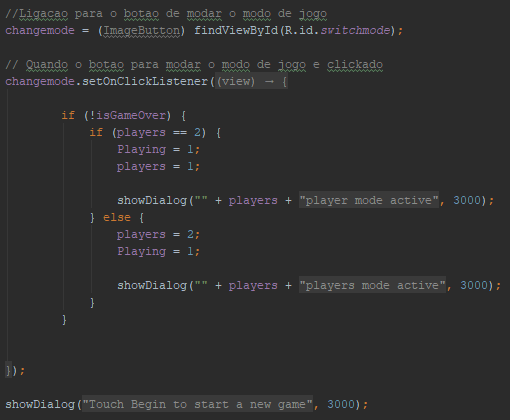
Na função ‘OnCreate’ vamos buscar os dados que vêm no intent e guardar essa informação para sabermos as definições do jogo.

Fazemos as ligações das variáveis criadas e dos objetos na layout e no botão de começar o jogo vamos fazer com que ele chame a função de começar o jogo quando este e clicado.



Para o botão para mudar de modo o que fazemos, tirando a ligação ao objeto, é uma verificação para sabermos no modo em que estamos e assim mudar-mos entre os 2 sem problemas. Esta mudança não pode ser feita caso o jogo já tenha acabo. Quando mudamos de modo é nos mostrada uma mensagem no ecrã a dizer que modo está agora ativo, para o utilizador saber da mudança.

Contudo quando criamos esta atividade e mostrada uma mensagem a dizer o que fazer para iniciar o campo de jogo.

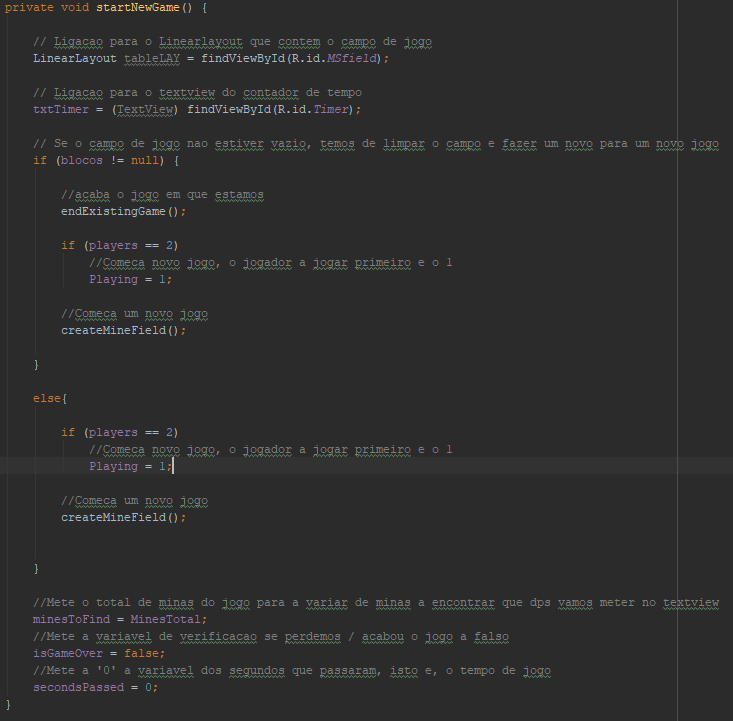


A função para começar um novo campo de jogo irá verificar se já existe um jogo a decorrer ou terminado, esta verificação é feita com o array dos blocos, caso este esteja a nulo é porque ainda não existe nenhum jogo criado anteriormente, por isso apenas precisamos de comecar um novo campo de jogo.

No caso de já existir dados no array de blocos, vamos fazer com que o jogo seja acabado, e criar um novo campo de jogo.

Em ambos os casos sempre que iniciamos um jogo e este esteja no modo de 2 jogadores é posto o primeiro jogador para ir jogar na próxima vez que o jogo iniciar o novo campo.

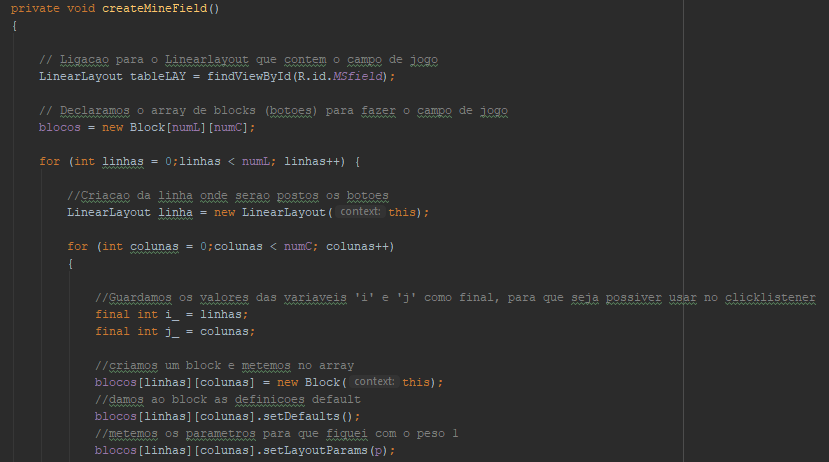
Já as variáveis de controlo serão então reiniciadas, as bombas a encontrar igual ao de bombas totais que iram haver no campo, o fim de jogo a false e o tempo que passou a 0, pois este e um jogo novo.



A função para a criação do campo de jogo começa por criar uma ligação com a layout que ira receber o campo de jogo. Declaramos então o array dos blocos com as dimensões recebidas no intent.

Usamos então 2 ‘for’ para irmos criando linhas que serão linearlayouts onde metemos os blocos, cada linha terá o numero de blocos igual ao numero de colunas que queremos.

Todos os blocos criados terão então as definições base dadas por a função para o efeito e os parâmetros para que assim todos tenham o peso 1.



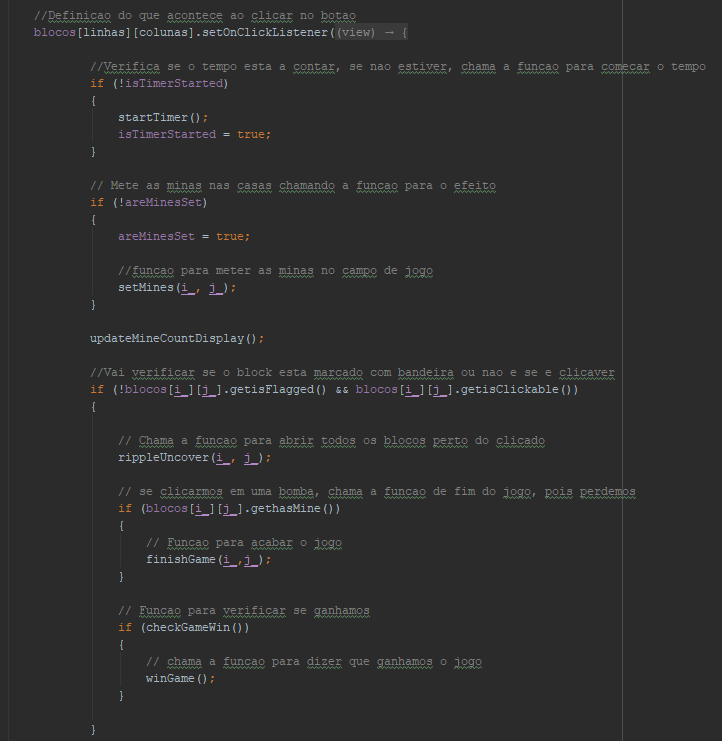
Depois de criar o bloco temos de lhe atribuir o que faz quando este recebe um click, sempre que o jogo e iniciado e o primeiro click e feito num bloco, este irá começar a contar o tempo, pois verifica se este já está ou não a contar, caso ainda não estejam as bombas no campo, também as introduz, no entanto envia a linha e coluna em que clicamos para que não seja metida uma bomba no bloco.

Quando e clicado um bloco este fará o update das bombas a encontrar no layout.

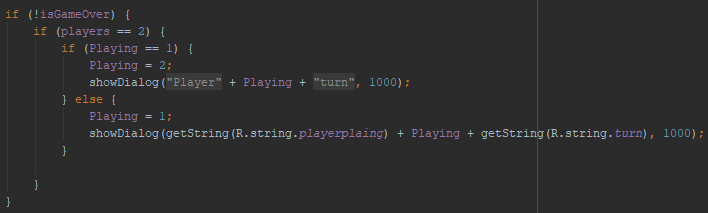
Também e verificado, ao clicar num bloco, se este não tem bandeira e se esta ativo, pois no caso de ter bandeira o sistema não fará nada tal como se não estiver ativo.

No caso de esta verificação ser verdadeira vamos abrir o bloco e todos os outros perto de si, mas com algumas restrições.

Verificamos se o bloco é ou não uma bomba, caso seja, chamamos a função para terminar o jogo, mas no caso não ser, verificamos o jogador ganhou ou não, no caso de ter ganho e chamada a função para dizer ao jogador que ganhou.



Quando o jogo não acaba vamos verificar se estamos no modo de 2 jogadores para que seja então alterado o jogador que vai fazer a próxima jogada e mostrar a mensagem a dizer quem a irá fazer.

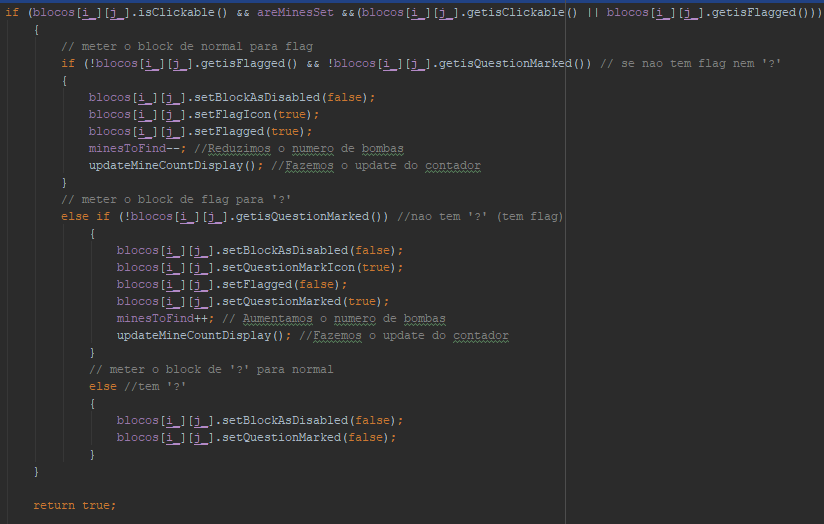


Também nos blocos temos o longclick, que fará o nosso bloco mudar de atributos.

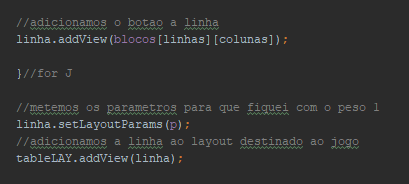


Quando esta ação e feita, o jogo vai verificar se o bloco está num de três estados, normal, flag ou ‘?’.

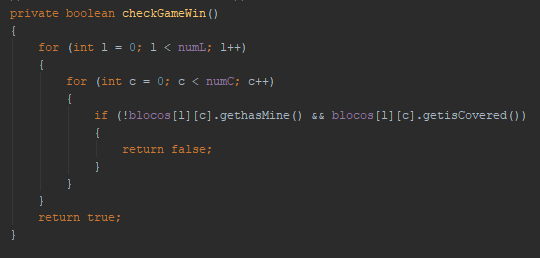
Para que esta ação seja feita o bloco tem de estar ativo, as bombas postas e ter bandeira, isto irá então verificar qual o estado do bloco e leva-lo para o próximo, normal para bandeira, bandeira para ‘?’ e ‘?’ para normal novamente, mudando assim os atributos do mesmo.



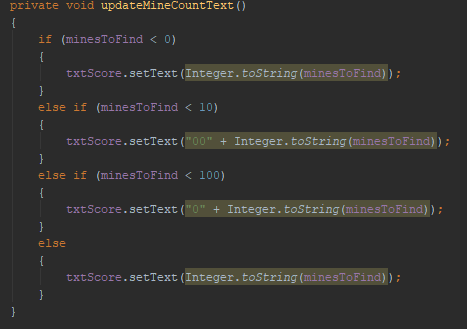
Ao acabar de programar o bloco, este é posto numa linha e quando essa linha tem o numero de blocos igual ao de colunas pretendidas, vai meter a linhas com os parâmetros que criamos e adiciona-la ao layout do campo de jogo.



A função para verificar se ganhamos o jogo, corre o array de blocos e caso encontre um bloco que não contenha bomba e ainda esteja por abrir, retorna ‘false’, o que diz ao programa que ainda existem blocos por abrir, no caso de todos os blocos que não contêm bomba já estiverem abertos retorna ‘true’ e assim saber que o jogo acabou.

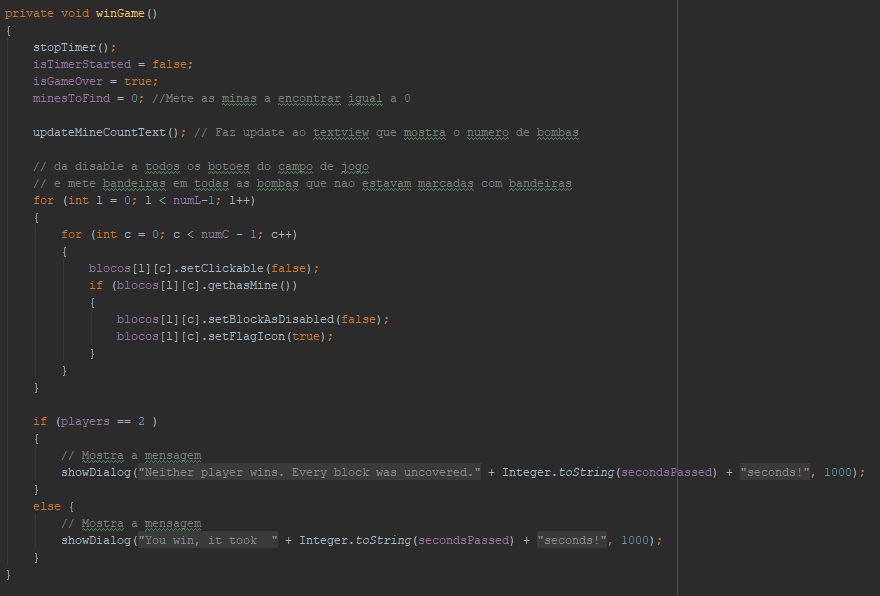


Esta função serve para quando metemos ou tiramos uma bandeira o contador atualizar com o numero que falta, fazendo com que o aspeto do texto seja sempre 3 dígitos, a não ser que este vá para uma contagem negativa.

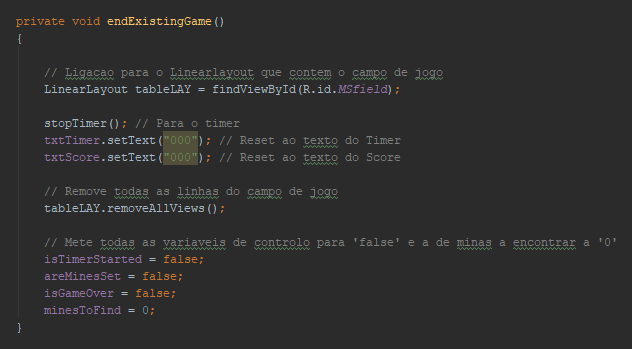


Esta Função quando o jogo é ganho, está para a contagem de tempo, da o jogo como acabo, mete todos os blocos desativos e mete bandeiras em todos os blocos que contenham bombas.

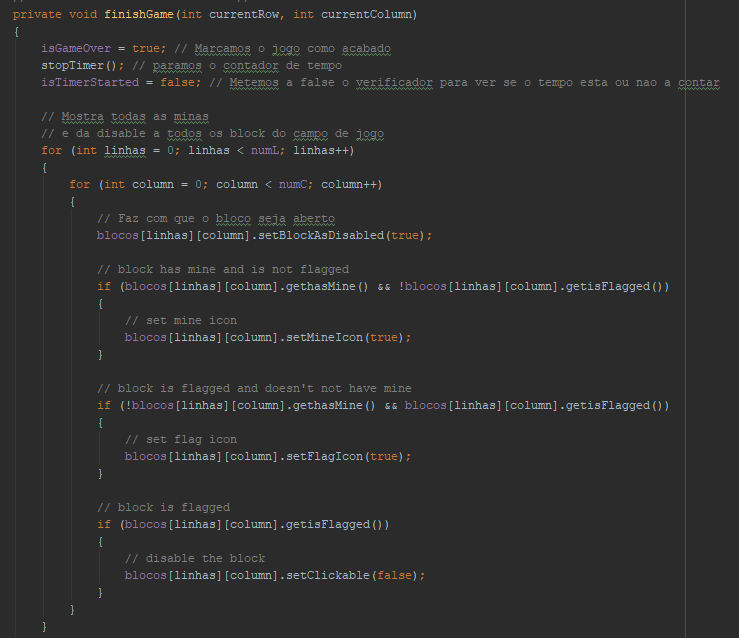
No fim mostra uma mensagem a dizer que o jogo terminou, no caso de dois jogadores, diz que não houve vendedor, se não, diz que o jogador ganhou e o tempo que levou até acabar o jogo.



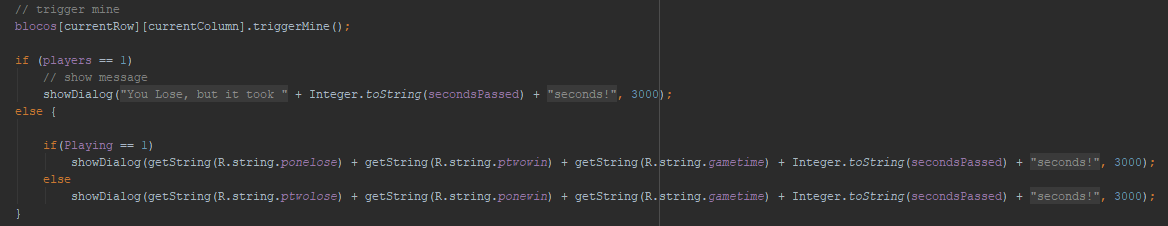
Esta função faz o jogo acabar e limpa o conteúdo do layout, isto é, vai meter os textos a ‘0’, apagar o conteúdo do layout do campo de jogo e meter todas as variáveis de verificação a ‘false’ ou ‘null’.



A função para ‘finishgame’ é chamada quando o jogo e perdido, esta desativa todos os blocos e abre todos os blocos, deixando as bandeiras que estavam em lugares errado e mostrando as bombas.



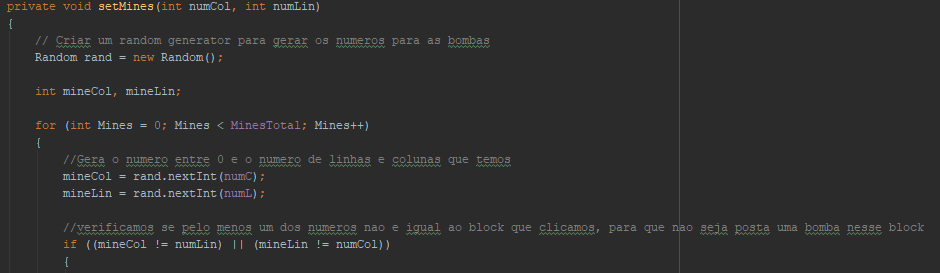
Apos isto a função ainda mete a bomba clicada como explodida, e vai mostrar uma mensagem, no caso do modo de 1 jogador esta diz que perdeu e o tempo de jogo que foi, já no modo de 2 jogadores diz qual ganhou e perdeu e o tempo de jogo.



Esta função é encarregue por meter as bombas no campo de jogo após o primeiro click,

As bombas são postas em posições geradas aleatoriamente com um random no intervalo de ‘0’ ao numero de colunas e linhas.

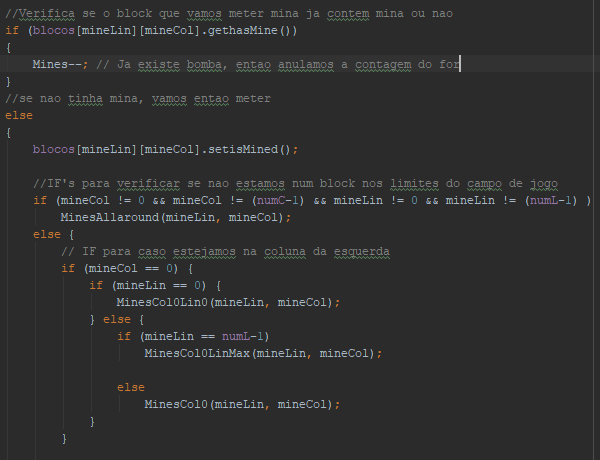
Antes de ser colocada a bomba é feita uma verificação para que não seja colocada no bloco em que o jogador carregou.



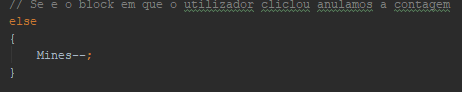
Sempre que é colocada uma bomba, verificamos se já não existe uma no bloco gerado, caso já exista uma bomba no bloco, anulamos a contagem do for, pois a bomba não foi posta.

Sempre que e metida uma bomba, são feitas varias verificações para sabermos se a bomba foi colocada nas extremidades do campo de jogo, pois os blocos a volta diferem sempre de onde esta e colocada, a ser que não esteja nas extremidades.

Esta verificação e feita para que nos blocos perto da bomba seja feita a contagem de bombas junto a estes, sendo assim temos de saber onde esta foi colocada para que não haja problemas nos limites do array.

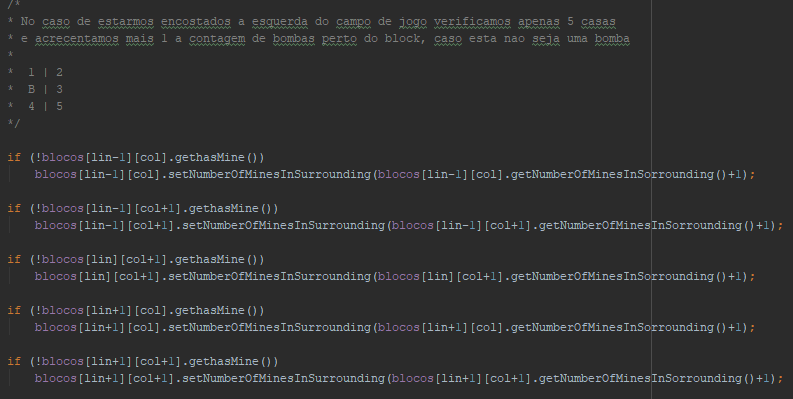


Quando o bloco é igual ao utilizador também temos de anular a contagem pois a bomba não é colocada.



Estas funções de contagem o que fazem e perceber onde a bomba foi colocada e saber os blocos que existem a volta e então colocar o numero de bombas que existe perto deles.

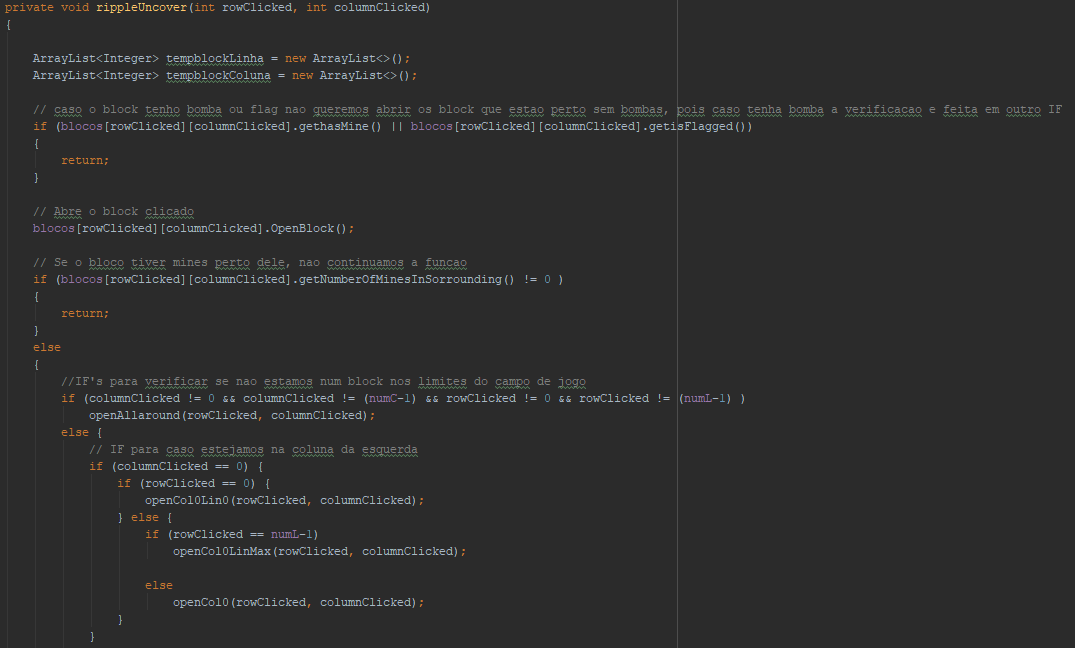
Na imagem podemos ver um exemplo quando a bomba é colocada na extremidade esquerda do campo de jogo.



A função ‘rippleUncover’ segue o mesmo principio que a função que usamos para meter as minas no campo, no entanto, esta ira abrir todos os blocos que não sejam bombas e que estejam perto de um bloco que contenha ‘0’ bombas junto a ele.

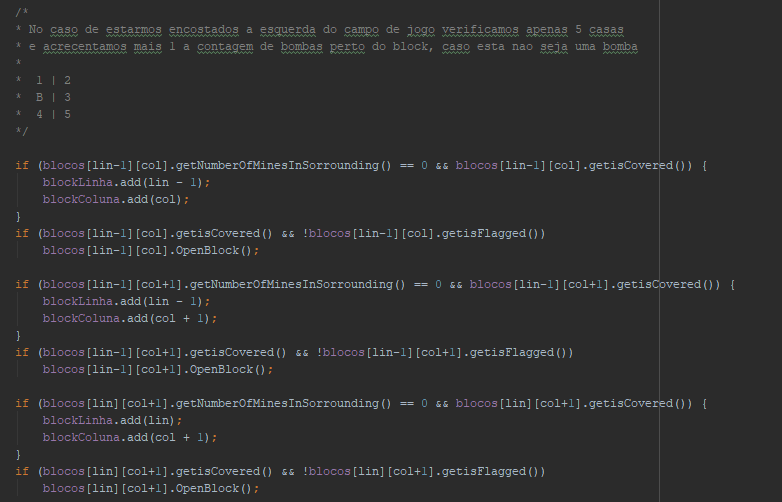
Sempre que clicamos em um bloco que tenha ‘0’ bombas junto a ele, esta função ira verificar se junto a este existem outros com ‘0’ bombas junto e guarda a localização destes em arrays para que assim seja feita a verificação se também estes tem outros assim.

Para fazer este processo o bloco clicado vai guardar os dados dos ‘0’ que existam a sua volta num array criado na função, caso o bloco tenha bombas junto a ele, este processo não e feito, e apenas e aperto.

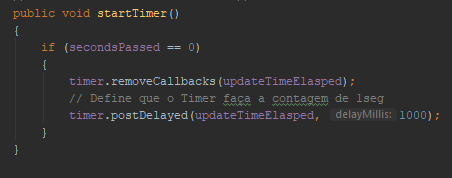


Quando existem outros ‘0’ o que é feito é copiar os array para outros temporários e então fazer a verificação feita no primeiro bloco a estes que foram guardados. E no fim de correr os arrays eles são limpos para que não haja repetição. Os blocos não podem estar apertos para que sejam guardados, caso estejam apertos, quer dizer que este processo já passou por eles.

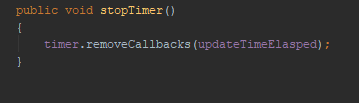
Este processo e feito da mesma maneira do de contagem de bombas juntos aos blocos, no entanto este verifica se os blocos a voltar são ‘0’ e ainda não foram aberto, e guarda-os para que seja feita a verificação neles, caso isto não seja verdade os blocos são apenas abertos.



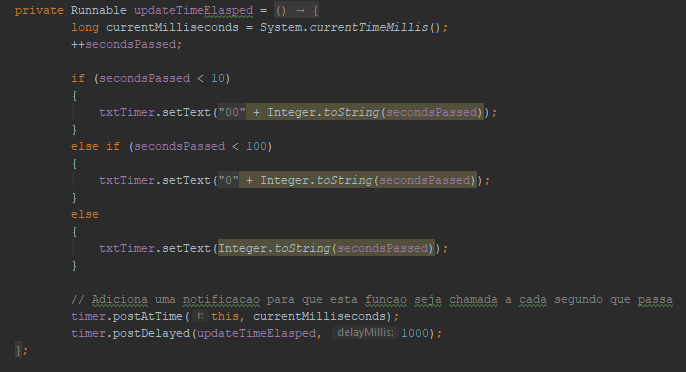
Esta função vai fazer começar o handler para que seja feita a contagem do tempo de jogo, e atualizar o contador de 1 em 1 segundo, parando o processo caso este já estivesse a ser feito primeiro.



Esta função apenas para as chamadas do handler para que assim o contador não avance.

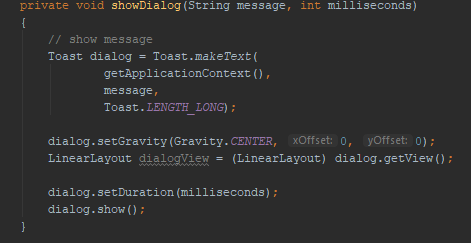


O runnable e chamado sempre que passa o tempo que metemos no timer, em milissegundos, este atualiza então o textview que nos mostra o tempo de jogo no layout da actividade.



Esta funcao encarrega-se de crair as mensagens que são criadas no ecran com o texto e o tempo recebido, isto e feito criando um Toast, que é uma mensagem como popup.

Esta e depois configurada a aparecer no sempre no ecran, neste caso, com o texto e o tempo que e recebido.



# Conclusão

Neste trabalho abordámos o assunto que consistia em um jogo, minesweeper, para um sistema android, com a possibilidade de 1 ou 2 jogadores.

Não cumprimos todos os objetivos a 100%, uma vez que alguns deles não sabíamos como fazer funcionar de como a implementar no projeto final, no entanto conseguimos fazer varias outras coisas que não tínhamos pensado de inicio.

Este trabalho no entanto foi bastante importante para se aprender que uma aplicação, para um sistema android, tem de ser toda ela muito bem pensada e estruturada para que não haja problemas com o seu funcionamento e que não seja preciso depois já estar quase completa, levar-nos a um ponto em que haja necessidade de grandes modificações, também temos de ter em atenção as especificações dos dispositivos pois todos são diferentes e não aguentam a mesma sobrecarga de processamento.