```java
package com.games.tictactoefree;

import ...

/* BackgroundSoundService for playing background music in game*/
5 usages
public class BackgroundSoundService extends Service {

    3 usages
    private MediaPlayerSound mpSound;

    @Override
    public IBinder onBind(Intent intent) { return null; }

    @Override
    public void onCreate() {
        super.onCreate();
        mpSound = MediaPlayerSound.getSingleton( ctx: this);
    }

    5 usages
    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        mpSound.startBackGroundSound();
        return START_NOT_STICKY;
    }

    @Override
    public void onDestroy() {
        mpSound.stopBackGroundSound();
```

```java
package com.games.tictactoefree;

import ...

10 usages
public class CircleSlot extends View{
    2 usages
    private int mRow, mColumn, mWidth, mHeight, mMaxWidth, mMaxHeight, mId;
    1 usage
    private Paint mPaint = new Paint();
    1 usage
    private Paint mBorderPaint = new Paint();
    3 usages
    private int mColor = Color.GRAY;
    //private Context mContext;

    1 usage
    public CircleSlot(Context context, int width,int height , int row , int column,int id) {
        super(context);
        mBorderPaint.setColor(Color.BLACK);
        //mContext = context;
        mId = id;
        mRow = row;
        mColumn = column;
        mWidth = width - row;
        mHeight = height- column;
        mMaxWidth = mWidth / mColumn;
        mMaxHeight = mHeight / (mRow);
    }
```

```
2 usages
public class LineSlot extends View {
    2 usages
    private int  mColumn, mWidth,  mMaxWidth;
    2 usages
    Paint paint = new Paint();

    1 usage
    public LineSlot(Context context, int width,int height , int row , int column) {
        super(context);
        paint.setColor(Color.rgb( red: 66,  green: 33,  blue: 00));
       // paint.setStrokeWidth(1);
        mColumn = column;
        mWidth = width;
        mMaxWidth = mWidth / mColumn;

    }

    @Override
    public void onDraw(Canvas canvas) {

        canvas.drawLine( startX: 0, startY: 0, mMaxWidth, stopY: 0, paint);
        /*if(mWidth > mHeight){
            canvas.drawLine(0,0, mMaxWidth,0, paint);
        }

        else{
            canvas.drawLine(0, mMaxHeight, mMaxHeight, mMaxHeight, paint);
             canvas.drawLine(mMaxHeight, 0, mMaxHeight, mMaxHeight, paint);
```

```java
package com.games.tictactoefree;

import ...

2 usages
public class LineSlotVertical extends View {
    3 usages
    private int mHeight;
    2 usages
    Paint paint = new Paint();

    1 usage
    public LineSlotVertical(Context context, int width,int height , int row , int column) {
        super(context);
        paint.setColor(Color.rgb( red: 66,  green: 33,  blue: 00));
        mHeight = height;
    }

    @Override
    public void onDraw(Canvas canvas) {


        //if(mWidth > mHeight){
            canvas.drawLine( startX: 0, startY: 0,  stopX: 0,mHeight, paint);
        //}

        /*else{
            canvas.drawLine(0, mMaxHeight, mMaxHeight, mMaxHeight, paint);
             canvas.drawLine(mMaxHeight, 0, mMaxHeight, mMaxHeight, paint);
        }*/
```

```java
public LinearLayout initializeUI(Context ctx) {
    TicTacToeEngine engine = TicTacToeEngine.singleton();
    mRow = engine.getNumRows();
    mColumn = engine.getNumColumn();
    circleSlots = new View[mRow][mColumn];
    LinearLayout pre_midd = new LinearLayout(ctx);
    pre_midd.setOrientation(LinearLayout.HORIZONTAL);
    return pre_midd;

}

1 usage
public View createSlots( int width, int height, Context ctx, View.OnClickListener slotEventHandler ) {
    //Context ctx = mCurrentGame.getContext();

    LinearLayout mainLayout = new LinearLayout(ctx);
//  mainLayout.setBackgroundResource(R.drawable.round_borders);
    for(int columnindex = 0; columnindex<mColumn; columnindex++) {
        LinearLayout linearlayout = new LinearLayout(ctx);
        linearlayout.setOrientation(LinearLayout.VERTICAL);
        linearlayout.setId(columnindex);
        LinearLayout.LayoutParams params1 = new LinearLayout.LayoutParams(LayoutParams.WRAP_CONTENT,LayoutParams.MATCH_PAR
        linearlayout.setLayoutParams(params1);
        LinearLayout linearLayoutForVerticalLine = new LinearLayout(ctx);
        linearLayoutForVerticalLine.setOrientation(LinearLayout.VERTICAL);
        LinearLayout.LayoutParams params2 = new LinearLayout.LayoutParams(LayoutParams.WRAP_CONTENT,LayoutParams.MATCH_PAR
        linearLayoutForVerticalLine.setLayoutParams(params2);

        //addColumn
```

```java
2 usages
public void updateUIstate(View.OnClickListener slotEventHandler, int currentPlayerType) {

    for(int columnindex = 0; columnindex<mColumn; columnindex++) {

        ArrayList<View>ar = arrayOfSlots.get(columnindex);
        for(int rowindex = 0 ;rowindex<mRow;rowindex++){

            CircleSlot slot = (CircleSlot) ar.get(rowindex);
            TicTacToeEngine engine = TicTacToeEngine.singleton();

            int matrixValue = engine.getStorageValueAtPosition(rowindex, columnindex);
            if(matrixValue == TicTacToeEngine.FirstPlayer) {
                slot.setColor(Color.RED);
                slot.setOnClickListener(null);
                slot.invalidate();
            }
            else if(matrixValue == TicTacToeEngine.Secondplayer) {
                slot.setColor(Color.GREEN);
                slot.setOnClickListener(null);
                slot.invalidate();
            }
            else {
                slot.setColor(Color.BLACK);
                if(currentPlayerType != 2)slot.setOnClickListener(slotEventHandler);
                else slot.setOnClickListener(null);
```

```
2 usages
public void destroySession() {
    _instance = null;
    System.gc();
}

1 usage
public ArrayList<Integer> getRandomSlots() {
    View vm = null;
    int nTry  = 0;
    boolean isfound = false;
    ArrayList<Integer>returnMatrix = new ArrayList<~>();
    while(true) {
        if(nTry < 10) {
            System.out.println("Tryies");
            Random r = new Random();
            int randomslot = r.nextInt( bound: 8);
            vm = (CircleSlot)numerOfSolts.get(randomslot);
            CircleSlot slot = (CircleSlot) vm;
            LinearLayout ll = (LinearLayout)slot.getParent();
            int columnSelected = ll.getId();
            int currentRow = TicTacToeUI.singleton().determineSelectedRow(vm, columnSelected);

            //TickTakToeEngine.singleton().printCurrentStorage();
            if(TicTacToeEngine.singleton().getStorageValueAtPosition(currentRow, columnSelected) == -1) {
                returnMatrix.add(currentRow);
                returnMatrix.add(columnSelected);
```

```java
public ArrayList<Integer> getEmptyRendomFromStorage() {
    System.out.println("TRY TO FINd EMPTY location");
    ArrayList<Integer>arList = new ArrayList<~>();
    //int localMatrix[][] = TickTakToeEngine.singleton().getCurrentStorage();
    for(int columnindex = 0; columnindex <mColumn; columnindex++ ) {
        for(int rowindex = 0; rowindex < mRow; rowindex++){
            if(TicTacToeEngine.singleton().getStorageValueAtPosition(rowindex, col
                arList.add(rowindex);
                arList.add(columnindex);
                return arList;
            }

        }
    }

    return arList;

}
2 usages
public int determineSelectedRow(View v, int currentColumn) {
    int selectedRow = -99;
    selectedRow = v.getId();
    System.out.println("d" + selectedRow);
    selectedRow  = selectedRow  - 100;
    return selectedRow;
}
```

```java
package com.games.tictactoefree;

import ...


8 usages
public class TicTacToeUI {
    8 usages
    int mRow;
    8 usages
    int mColumn;
    2 usages
    ArrayList<ArrayList<View>> arrayOfSlots = new ArrayList<ArrayList<View>>();
    2 usages
    ArrayList<View>numerOfSolts = new ArrayList<~>();
    2 usages
    private View circleSlots[][] = null;
    4 usages
    public static TicTacToeUI _instance;
    3 usages
    public static TicTacToeUI singleton() {
        if(_instance == null) {
            _instance = new TicTacToeUI();
        }
        return _instance;
    }

    1 usage
    public LinearLayout initializeUI(Context ctx) {
        TicTacToeEngine engine = TicTacToeEngine.singleton();
```