

```

;X11 X;

void setup() {
  size(600, 600);
  noStroke();
  background(255);
  ellipseMode(CENTER_DIAMETER);
  framerate(40);
  cursor(CROSS);
  smooth();

  X = new cell(1,width/2,height/2, width); //FirstCell
}

void loop() {
  //background(255);
  fill(255,20);
  rect(0,0,width,height);
  for(int i=0; i<1; i++) X.drawCell(); //DrawMany Cells per Loop
}

//This is the only object. It's used with recursivity.

class cell{
  int actualCell; //the parent control the state of his children
  int posX, posY; //the center of the block
  int max; //the width of the block (just max is ok as they're square)
  cell[] grid; //the array of children (won't use 2D array)

  cell(int sub, int posX, int posY, int max){
    this.posX = posX;
    this.posY = posY;
    this.max = max;

    if(sub>1) divideCell(sub);
    else grid = new cell[1];

    actualCell = 0;
  }

  boolean drawCell(){
    boolean tst = false;
    if(grid.length==1)
    {
      drawBlock();
      accidentalCell();
      tst = true;
    }
    else {
      if(grid[actualCell].drawCell() )
        if(inc() ) tst = true;
    }
    return tst;
  }

  void drawBlock(){
    fill(0);
    ellipse(posX, posY,0.95*max,0.95*max);
    fill(255);
    ellipse(posX, posY,0.85*max,0.85*max);
    fill(0);
    ellipse(posX, posY,0.28*max,0.28*max);
    fill(255);
    ellipse(posX, posY,0.25*max,0.25*max);
    fill(0);
    ellipse(posX, posY,0.15*max,0.15*max);
  }

  void divideCell(int sub){
    grid = new cell[sub*sub]; //Create the children's array
    if(sub>1){ //If it should be divided in more than one cell
      for(int i=0; i<sub; i++) //Rows
        for(int j=0; j<sub; j++) //Cols
          grid[i*sub+j]=new cell(1, (posX-(max/2)+(max/sub)*(i+1)-(max/(2*sub)), (posY-(max/2)+(max/sub)*(j+1)-(max/(2*sub)),max/sub));
    }
  }

  void accidentalCell(){
    if(random(100)>95) //It happens sometimes
    {
      int sub = (int)random(3)+1; //Number of divisions
      divideCell(sub);
    }
  }

  boolean inc(){
    boolean tst = false;
    actualCell++;
    if(actualCell>=grid.length)
    {
      actualCell =0;
      tst =true;
    }
    return tst;
  }
}

```