BAB 5
GRAFIK VEKTOR

5.1 Pendahuluan

Grafik vektor banyak sekali digunakan pada pembuatan game-game 2 dimensi dimana terdapat banyak method yang sudah disediakan oleh Java seperti ketika menangani pengujian tabrakan, menampilkan bound berupa rectangle dan masih banyak lagi. Grafik vektor merupakan pelajaran dasar yang harus dipelajari sebelum memasuki pembahasan grafik bitmap. pada modul ini juga dibahas masalah tentang penanganan grafis full screen dimana game yang akan dibuat dapat dimainkan dalam satu layar penuh.

5.2 Tujuan

Setelah mempelajari modul ini peserta diharapkan dapat:

- Mengetahui cara menggunakan interface Shape.
- Menggambar garis menggunakan class Graphics2D.
- Mengetahui perbedaan antara Rectangle2D dan RoundRectangle2D.
- Menggambar menggunakan Arc2D.
- Memahami fungsi dari Polygon.
- Membuat tampilan mode Full-Screen
- Menjalankan dan menutup mode Full-screen

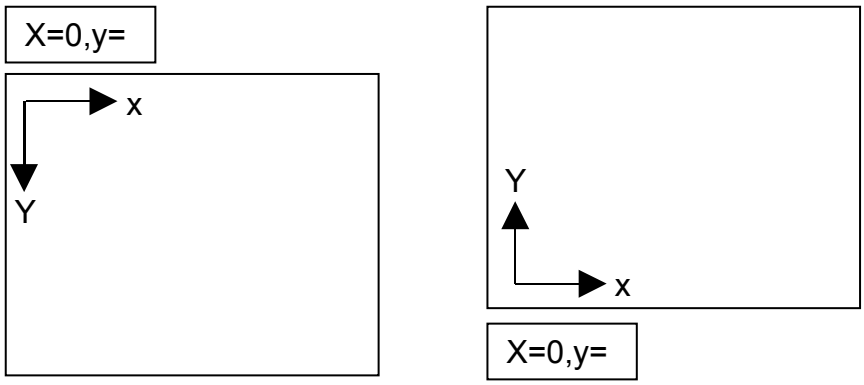
5.3 Grafik Vektor

Java 2 Dimensi yang tepatnya terdapat pada Class Graphics2D yang berada didalam java.awt.* menyediakan banyak fitur untuk bekerja dengan grafik vektor 2D, yang memungkinkan pengguna untuk menggambar bentuk

persegi panjang, poligon dan bentuk-bentuk lain dengan mudah. Class-Class yang ada didalam Graphics2D tidak hanya menggambar saja tetapi juga dapat menggeser, merotasi, dan menskala bentuk.

5.4 Koordinat di Java

Berbeda dengan bahasa pemrograman lain, java mempunyai dua sistem koordinat yang pertama user space adalah koordinat yang digunakan oleh program java, semua yang berbentuk geometri di petakkan di koordinat ini. Yang kedua yaitu device space adalah sistem koordinat device yang tergantung pada output device yang digunakan.Bisanya konversi dari device space ke user space dilakukan secara otomatis ketika proses rendering. Gambar dibawah ini menunjukan perbeadaan antara user space dan device space.

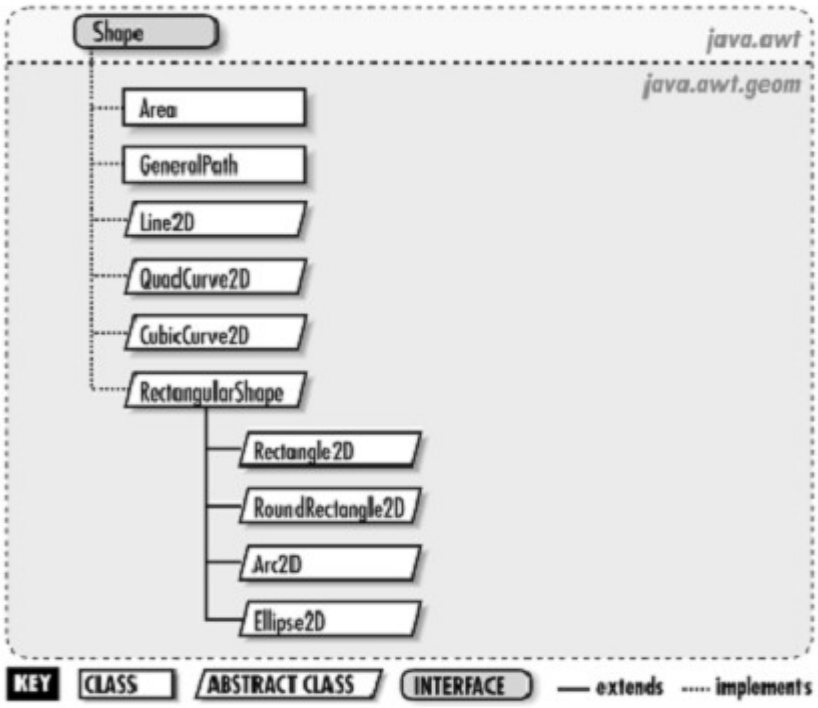


Gambar 5.1 Sistem Koordinat

5.5 Interface Shape

Sebelum ke pembahasan yang lebih dalam lagi sebaiknya kita mengetahui terlebih dahulu tentang Interface Shape. Interface Shape ini terdapat dalam package java.awt.* yang mendefinisikan tentang grafik primitif seperti titik, garis, sudut, persegi panjang, kurva, dan elips seperti terlihat pada

gambar dibawah. Class-class seperti Rectangle menggunakan nilai integer, sedangkan Rectangle2D menggunakan float dan double.



Gambar 3.16 Implementasi Interface Shape

5.6 Bekerja dengan Line2D

Tipe yang paling dasar dari shape adalah garis, dibuat dengan class Line2D.Float, untuk membuat garis harus menuliskan secara spesifik koordinat x dan y untuk titik awal dan akhir. Sebagai contoh:

```
Shape garis = new Line2D.Float(0,0,100,100);
```

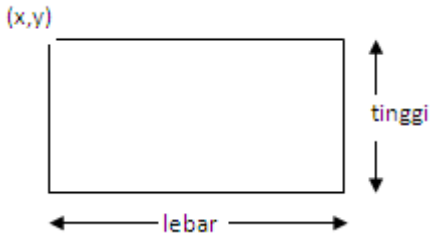
Kode diatas, menentukan posisi garis dibuat mulai dari koordinat (0,0) sampai dengan (100,100).

Line2D.Float adalah class mempunyai alternatif konstruktor yang dapat menerima object Point2D sebagai parameter. Point2D juga sebuah Abstract class dengan inner class Float dan Double untuk itu harus menggunakan Point2D.Float atau Point2D.Double pada awal pembuatan object. Dibawah ini adalah contoh yang sama menggambarkan garis menggunkan object point2D.

```
Point2D awal=new Point2D.Float(0,0);
Point2D akhir=new Point2D.Float(100,100);
Shape garis = new Line2D.Float(awal, akhir);
```

5.7 Bekerja dengan Rectangle2D & Ellipse

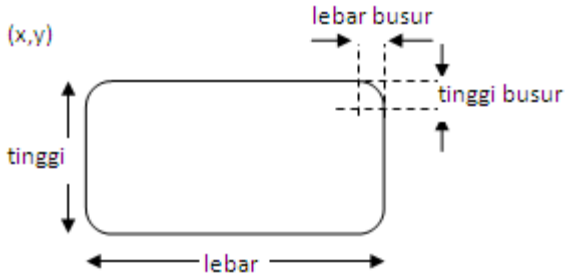
Rectangle membutuhkan titik awal (x, y) dan (lebar, tinggi). Dibawah ini sebuah rectangle dimulai pada titik (20,20) dengan lebar 50 dan tinggi 30. Bentuk dari rectanggle dapat dilihat seperti gambar disamping.



Gambar 3.17 Rectanggle2D

```
Shape rect = new Rectangle2D.Float(20,20,50,30);
```

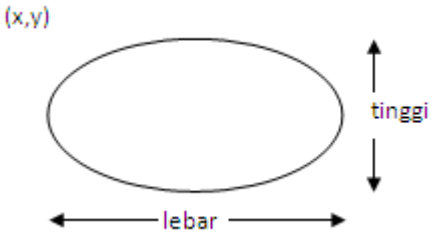
Java juga menyediakan class RoundRectangle2D yang memungkinkan untuk menggambar dengan lengkungan pada setiap sudut. Constructor pada class ini menambahkan dua parameter lagi untuk menentu masing-masing sudut.



Gambar 3.18 RoundRectanggle2D

```
Shape round_rect = new RoundRectangle2D.Float(20,20,50,30,10,10);
```

Ellipse2D tidak jauh berbeda dengan Rectangle2D hanya saja jika batas dari rectangle adalah persegi maka ellipse adalah lingkaran. Bentuk dari ellipse dapat dilihat pada gambar disamping.



Gambar 3.19 Ellipse2D

```
Shape ellipse = new Ellipse2D.Float(20,20,50,30);
```

5.8 Bekerja dengan Arc2D

Tipe dari shape yang sangat beguna adalah Arc2D. Yang perlu diperhatikan dalam pembuatan Arc2D adalah: Sudut 0º dimulai dari sebelah kanan atau dari arah jam 3. Perlu diingat bahwa penentuan sudut berlawanan dengan arah jarum jam. Ada tiga macam parameter akhir yang digunakan pada Arc2D yaitu:

OPEN

Konstanta ini mereperesentasikan bentuk busur terbuka yang berupa sebuah garis kurva bagian luar dari ellipse.

PIE

Konstatanta ini merepresentasikan sebuah busur yang merupakan potongan dari pie dimana garis kurva merupakan garis luar dari elipse dan garis lurus dari ujung busur ke titik tengahnya.

CHORD

Konstanta ini merepresentasikan sebuah busur dimana terdapat garis lurus yang menghubungkan antara kedua ujung busur.

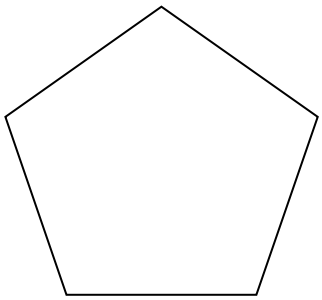
```
Shape arc1 = new Arc2D.Float(20,20,50,30,90,90,Arc2D.OPEN);
//atau:
Shape arc1 = new Arc2D.Float(20,20,50,30,90,90,Arc2D.CHORD);
//atau:
Shape arc1 = new Arc2D.Float(20,20,50,30,90,90,Arc2D.PIE);
```

5.9 Bekerja dengan Polygon

Kelas polygon sedikit berbeda dengan Rectangle karena Polygon mengijinkan untuk mendefinisikan bentuk sesuai dengan keinginan sendiri dengan menggunakan pasangan nilai X dan Y. Untuk membuat sebuah polygon cukup dengan sebuah titik tunggal atau sebuah polygon dengan empat titik untuk menyalin class Rectangle.

```
int titikX[] = {200,100,130,270,300};
int titikY[] = {50,150,270,270,150};
```

Ketika membuat sebuah polygon dengan cara ini, perhatikan array X dan Y harus berpasangan, maksudnya setiap nilai X harus berpasangan dengan setiap nilai Y. Terkadang cukup membingungkan ketika anda harus mendefinisikan bentuk polygon dengan menggunakan 2 buah array dari titik X dan Y, jadi sebaiknya mendesain polygon pada kertas atau editor grafik terlebih dahulu. Gambar dibawah menunjukkan desain dari polygon 5 sisi.



Gambar 5.2 Bentuk Gambar Koordinat Poligon

Melihat sebuah diagram dengan gambar akan sangat membantu, khususnya ketika mempunyai pekerjaan membuat sebuah polygon yang kompleks. Berikut ini adalah potongan kode pendefinisian polygone.

```
Polygon poligon= new Polygon(titikX, titikY, titikX.length);
```

Untuk menggambar sebuah bentuk, baik itu sebuah kotak, sebuah polygon atau yang lain, ada dua method pilihan yaitu method fill() untuk menggambar bentuk dengan warna didalamnya atau dapat menggunakan method draw() untuk menggambar garis luar / border dari sebuah bentuk dalam warna tertentu.

```
// Gambar Bentuk dengan didalamnya
g2d.fill(poligon);
// Gambar garis luarnya saja
g2d.draw(poligon);
```

Jangan lupa sebelum digambar seting warna pada Graphics2D sesuai dengan keinginan,agar lebih mudah dalam setting warna sebaiknya menggunakan class Color dengan pemanggilan attribut abstractnya yaitu jenis-jenis warna.

```
// Setting Warna menggunakan Class Color sebagai parameter
g2d.setColor(Color.RED);
```

5.10 Membuat Tampilan Mode Full-Screen

Mode Full-screen berada didalam object java.awt.GraphicsDevice, didalam singel atau multi-monitor dapat memanggil method getScreen-Devices pada `java.awt.GraphicsEnvironment` dan untuk single monitor dapat menggunakan getDefaultScreenDevices.

```
private GraphicsDevice device;
.........
.........
    GraphicsEnvironment environment =
        GraphicsEnvironment.getLocalGraphicsEnvironment();
    device = environment.getDefaultScreenDevice();
```

5.10.1 Display Mode

DisplayMode adalah class yang disediakan oleh java untuk mengenkapsulasi ukuran (tinggi, lebar dari monitor dengan satuan pixel), ketajaman bit (banyaknya bit per pixel), dan refresh rate (frekwensi monitor untuk kecepatan rata-rata dalam menampilkan citra). Untuk mengetahui display mode pada layar cukup dengan memanggil method getDisplayMode dan jika ingin menampilkan semua mode display yang didukung gunakan

method getDisplayModes. Sebelum mencoba untuk merubah display mode sebaiknya periksa terlebih dahulu apakah sistem mendukung atau tidak dengan memanggil method isDisplayChangeSupported jika meengembalikan nilai true berarti sistem operasi mendukung untuk merubah display mode.

```
....
....
if (device.isDisplayChangeSupported())
{
    device.setDisplayMode(displayMode);
}else{
    System.out.println("Sistem Operasi Tidak Mendukung Mode
DisplayChange");
    System.exit(0);
}
....
....
```

Untuk memeriksa apakah sistem operasi mendukung mode full-screen atau tidak gunakan method isFullScreenSupported akan mengembalikan nilai true apabila object GaphicsDevices mendukung mode Full-screen dan mengembalikan nilai false jika tidak mendukung mode tersebut.

```
public void setFullScreen(DisplayMode displayMode,JFrame frame) {
if(device.isFullScreenSupported()){
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      frame.setUndecorated(true);
      frame.setResizable(false);
      device.setFullScreenWindow(frame);

  if(displayMode != null && device.isDisplayChangeSupported()){
      try {
            device.setDisplayMode(displayMode);
      }catch (IllegalArgumentException ex) { }
  }


}else{
   System.out.println("System tidak mendukung Mode Full-screen");
}

}
```

Method setUndecorated() diberi nilai true untuk menghilangkan title bar dan bingkai pada frame, sedangkan method setResizeable() diberi nilai false agar screen tidak dapat diubah-ubah ukurannya, kemudian untuk merubah

tampilan full-screen menggunkan method setFullScreenWindow() dimana argumen yang dilewatkan adalah JFrame. Perlu diketahui bahwa JFrame yang dilewatkan di dalam method setFullScreen(.... , ...) diatas tidak boleh di setting setVisible(true) karena jika di setting demikian maka method FullScreen tidak dapat berjalan sesuai dengan keinginan.

Selanjutnya mendefinisikan beberapa DisplayMode atau ukuran resolusi yang akan digunakan untuk menentukan tinggi dan lebar layar yang diinginkan hanya dengan menentukan array dari resolusi yang telah disediakan, listing programnya adalah sebagai berikut.

```
private static final DisplayMode Mode_Pilihan[] = {
      new DisplayMode(1024, 768, 32, 0),
      new DisplayMode(1024, 768, 24, 0),
      new DisplayMode(1024, 768, 16, 0),

       new DisplayMode(800, 600, 32, 0),
      new DisplayMode(800, 600, 24, 0),
      new DisplayMode(800, 600, 16, 0),
      new DisplayMode(640, 480, 32, 0),
      new DisplayMode(640, 480, 24, 0),
      new DisplayMode(640, 480, 16, 0)
};
```

Array Mode_Pilihan diatas akan diperiksa oleh method periksaDisplayMode() yang melewatkan dua parameter yang bertujuan untuk memeriksa apakah sistem yang kita gunakan mendukung resolusi yang didefinisikan tersebut atau tidak.

```
public boolean perikasDisplayMode(DisplayMode mode1,DisplayMode
Mode2){
       if (mode1.getWidth() != mode2.getWidth() ||
            mode1.getHeight() != mode2.getHeight())
       {   return false; }
       if (mode1.getBitDepth() != DisplayMode.BIT_DEPTH_MULTI &&
           mode2.getBitDepth() != DisplayMode.BIT_DEPTH_MULTI &&
            mode1.getBitDepth() != mode2.getBitDepth())
       {   return false; }
       if (mode1.getRefreshRate() !=
            DisplayMode.REFRESH_RATE_UNKNOWN &&
            mode2.getRefreshRate() !=
            DisplayMode.REFRESH_RATE_UNKNOWN &&
            mode1.getRefreshRate() != mode2.getRefreshRate())
       {  return false; }
```

```
        return true;
}
```

Pada method periksaDisplayMode diatas akan mengembalikan nilai berupa boolean yang artinya jika tidak cocok dengan kondisi yang diperiksa diatas maka akan mengembalikan nilai false. Method ini akan dipanggil di dalam sebuah method setResolusiLayar() dimana method ini melewatkan satu parameter untuk menentukan index array yang akan di gunakan. Berikut ini adalah listing programnya.

```
public DisplayMode setResolusiLayar(int nilai){
      if(nilai >=9)nilai=0;
        DisplayMode sistemMode[] = device.getDisplayModes();
            for (int j = 0; j < sistemMode.length; j++) {
                if (perikasDisplayMode(Mode_Pilihan[nilai],
                        sistemMode[j])) {
                    return Mode_Pilihan[nilai];
                }
            }
        return null;
}
```

Seperti yang telah dijelaskan sebelumnya bahwa method getDisplayModes() yang berada didalam class GraphicsDevice akan memberikan nilai berupa array DisplayMode yang di dukung oleh sistem operasi yang kita gunakan dan selanjutnya akan di cross-cek apakah sama dengan DisplayMode yang kita tentukan pada parameter setResolusiLayar().

Setelah mendefinisikan variabel dan beberapa method diatas, sekarang kita membuat satu method lagi yaitu tutupScreen() yang akan dipanggil apabila user akan mengakhiri mode full-screen.

```
public void tutupScreen() {
        Window window = device.getFullScreenWindow();
        if (window != null) {
            window.dispose();
        }
        device.setFullScreenWindow(null);
}
```

Pada class GraphicsDevice terdapat method getFullScreenWindow() yang akan merepresentasikan jendela full-screen jika device dalam keadaan mode full-screen dan mengembalikan nilai *null* apabila tidak dalam keadaan mode full-screen. Method dispose() yang berada dalam class Window berfungsi untuk membebaskan semua native screen yang digunakan dan akan mengembalikan semua resource yang dipakai kepada sistem operasi. Jika semua berjalan lancar maka method setFullScreenWindow() diberi nilai null untuk menonaktifkan atau menutup mode full-screen.

5.10.2 Menjalankan dan Menutup Mode Full-Screen

Seperti yang kita ketahui bahwa class ScreenManager telah berhasil dibuat sekarang mari kita mencoba untuk memanggil dan menjalankan method-method yang ada didalam class tersebut seperti pada baris program berikut ini:

```
........
    private ScreenManager screen;

    public void () {
        screen = new ScreenManager();
        try {
            DisplayMode displayMode = screen.setResolusiLayar(1);
            screen.setFullScreen(displayMode,new PercobaanArc2d());
        }
        catch (Exception e){
             screen.restoreScreen();
        }
    }
.......
```

Pada potongan kode diatas kita membuat object dari DisplayMode yang digunakan untuk mengatur resolusi yang kemudian dilewatkan sebagai argument pertama dan class yang akan ditampilkan dalam mode full-screen yang dilewatkan pada argument kedua didalam method setFullScreen(). Untuk method restoreScreen() dipanggil apabila terjadi eror atau jika ingin menutup mode full-screen.