PRAKTIKUM 03

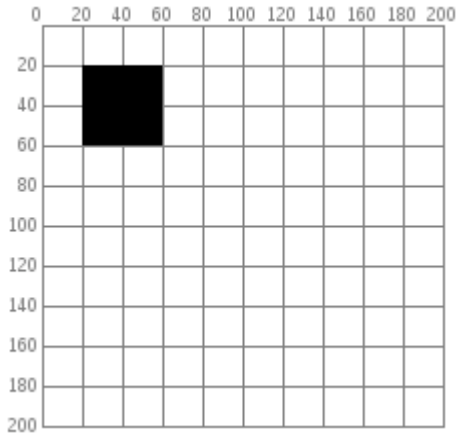
TRANSFROMASI

Tujuan Instruksional
Setelah mengikuti praktikum ini, mahasiswa diharapkan dapat untuk :
1. Memahami model transformasi objek
2. Membuat animasi tingkat dasar

Processing memiliki fungsi built-in yang membuatnya mudah bagi Anda untuk memiliki objek dalam sketsa gerak, memutar, dan tumbuh atau menyusut. Tutorial ini akan memperkenalkan Anda ke proses Translation, Rotate, dan scalling sehingga Anda dapat menggunakannya dalam sketsa Anda. Secara mendasar teknik animasi adalah Translation (pergeseran), Rotation (perputaran) dan Scalling (penskalaan), dengan mengkombinasi ketiga proses tersebut nantinya anda akan mendapatkan sebuah objek yang dianimasi.

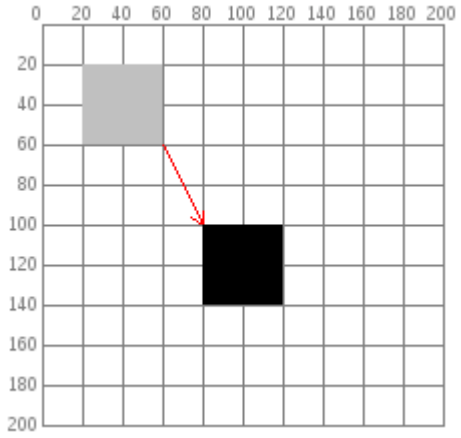
Translation

Seperti yang anda ketahui, layar yang disediakan processing adalah selayaknya sepotong kertas grafik. Jika kita ingin melakukan proses translasi sebenarnya terdapat dua pemikiran. Sebagai contoh adalah sebuah persegi sederhana dengan kode rect(20,20,40,40) seperti terlihat pada gambar 3.1.



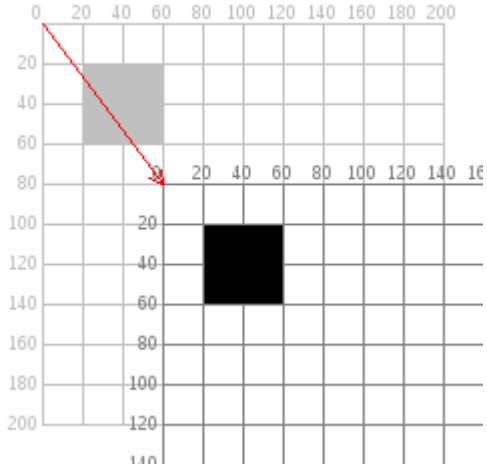
Gambar 3.1. Objek persegi sederhana

Jika anda ingin memindahkan persegi tersebut ke koordinat 60 unit ke kanan dan 80 unit kebawah, pada pemikiran pertama adalah bahwa anada langsung memindahkan objek kearea tersebut dengan cara menambahkan nilai x dengan 60 dan nilai y dengan 80, sehingga kodenya adalah rect(20+60,20+80,40,40) seperti terlihat pada gambar 3.2 dan program 3.1.



Gambar 3.2. Proses pemindahan objek dengan translasi

Tapi ada cara yang lebih menarik untuk melakukannya, yaitu memindahkan kertas grafik sebagai gantinya, perhatikan gambar 3.3. Jika Anda memindahkan kertas grafik 60 unit kanan dan 80 unit ke bawah, Anda akan mendapatkan hasil persis visual yang sama. Memindahkan sistem koordinat seperti ini disebut translation.



Gambar 3.3. Proses translasi sistem koordinat

Hal penting untuk diperhatikan dalam diagram sebelumnya adalah bahwa, persegi panjang yang bersangkutan, itu tidak bergerak sama sekali tetap sudut kiri atas adalah masih di (20,20). Bila Anda menggunakan transformasi, hal yang menarik tidak pernah mengubah posisi sistem koordinat. Berikut ini adalah kode yang digunakan untuk melakukan transformasi objek dan transformasi system koordinat.

Program 3.1

```
void setup()
{
  size(200, 200);
  background(255);
  noStroke();
  // draw the original position in gray
  fill(192);
  rect(20, 20, 40, 40);
```

```
  // draw a translucent red rectangle by changing the object coordinates
 fill(255, 0, 0, 128);
 rect(20 + 60, 20 + 80, 40, 40);
}
```

Program 3.2

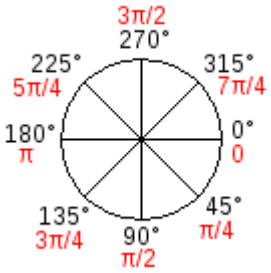
```
void setup()
{
 size(200, 200);
 background(255);
 noStroke();
 // draw the original position in gray
 fill(192);
 rect(20, 20, 40, 40);
  // draw a translucent blue rectangle by translating the grid
 fill(0, 0, 255, 128);
 pushMatrix();
 translate(60, 80);
 rect(20, 20, 40, 40);
 popMatrix();
}
```

Mari kita lihat kode terjemahan lebih terinci. pushMatrix () adalah fungsi built-in yang menyelamatkan posisi sistem koordinat. Translasi (60, 80) bergerak pada sistem koordinat 60 unit dan 80 unit kanan bawah. Rect (20, 20, 40, 40) mempoisikan persegi panjang di tempat yang sama awalnya. Ingat, objek tidak bergerak namun grid bergerak sebagai gantinya. Akhirnya, popMatrix () mengembalikan sistem koordinat awal sebelum Anda melakukan translasi.

Anda bisa melakukan translasi (-60, -80) untuk memindahkan grid kembali ke posisi semula. Namun, ketika Anda mulai melakukan operasi yang lebih canggih dengan sistem koordinat, akan lebih mudah untuk menggunakan pushMatrix () dan popMatrix () untuk menyimpan dan mengembalikan status daripada harus membatalkan semua operasi Anda.

Rotasi

Selain bergerak pada grid, Anda juga dapat memutar itu dengan fungsi rotate(). Fungsi ini memerlukan satu argumen, yang merupakan jumlah radian yang Anda ingin putar. Dalam Processing, semua fungsi harus dilakukan dengan mengukur sudut rotasi dalam radian (rad), bukan derajat. Ketika Anda berbicara tentang sudut dalam derajat, Anda mengatakan bahwa lingkaran penuh memiliki 360 °. Ketika Anda berbicara tentang sudut dalam radian, Anda mengatakan bahwa lingkaran penuh telah 2π radian. Berikut ini adalah diagram tentang bagaimana langkah-langkah Pengolahan sudut dalam derajat (hitam) dan radian (merah) gambar 3.4.



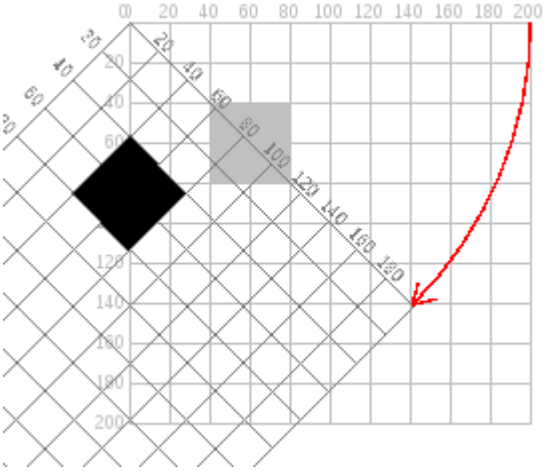
Gambar 3.4. Perbandingan sudut Radian dan Derajad

Karena kebanyakan orang berpikir dalam derajat, Processing memiliki built-in fungsi radian()yang mengambil sejumlah derajat sebagai argumen dan mengkonversi untuk Anda. Ini juga memiliki fungsi derajat() yang mengubah radian ke derajat. Sebagai percobaan pertama kita coba lakukan rotasi 45 derajad clockwise.

Program 3.3

```
void setup()
{
  size(200, 200);
  background(255);
  smooth();
  fill(192);
  noStroke();
  rect(40, 40, 40, 40);
pushMatrix();
  rotate(radians(45));
  fill(0);
  rect(40, 40, 40, 40);
  popMatrix();
}
```

Dari program 3.3, apa yang terjadi? Kenapa objek persegi yang diputar menjadi terpotong?. Jawabannya adalah, sebenarnya objek persegi diputar dengan titik pusat rotasi adalah (0,0) atau dengan kata lain dapat diartikan bahwa objek persegi tidak di-rotasi, namun yang dirotasi adalah system koordinatnya, seperti diperlihatkan pada gambar 3.5.



Gambar 3.5. Proses rotasi grid

Cara yang lain adalah dengan menggunakan lokasi lain untuk dijadikan sebagai pusat rotasi. Sebagai contoh, objek persegi yang diatas akan diputar 45 derajad dengan menggunakan titik pusat rotasi adalah pojok kiri atas. Cara yang digunakan adalah :

1. Translasi lokasi koordinat (0,0) kearah pusat rotasi, dalam hal ini adalah (40,40)
2. Gunakan fungsi rotate() dengan parameter 45 derajad untuk memutar grid
3. Gambarkan kembali objek dengan posisi awal adalah lokasi original grid
4. Kembalikan posisi grid ke awal dengan menggunakan fungsi popMatric()

berikut adalah kode yang dimaksudkan :

Program 3.4

```
void setup()
{
  size(200, 200);
  background(255);
  smooth();
  fill(192);
  noStroke();
  rect(40, 40, 40, 40);

  pushMatrix();
  // move the origin to the pivot point
  translate(40, 40);

  // then pivot the grid
  rotate(radians(45));

  // and draw the square at the origin
  fill(0);
  rect(0, 0, 40, 40);
  popMatrix();
}
```

Scalling

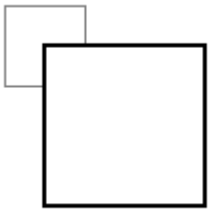
Transformasi yang terakhir adalah scalling, yang mana prosesnya adalah mengubah ukuran objek baik membesara atau pun mengecil. Lihat contoh

program berikut yang mengubah ukuran objek menjadi dua kali lebih besar dari originalnya.

Program 3.5

```
void setup()
{
  size(200,200);
  background(255);

  stroke(128);
  rect(20, 20, 40, 40);
  stroke(0);
  pushMatrix();
  scale(2.0);
  rect(20, 20, 40, 40);
  popMatrix();
}
```



Dari program diatas, maka seolah olah proses scalling memiliki titik prespektif dari pusat (0,0).

Tugas

1. Buatlah sebuah objek seperti berikut :

2. Buatlah program untuk melakukan proses scalling dari sembarang titik pusat skala

Tgl.	TUGAS PRAKTIKUM 03	Ttd. Dosen :
Teknik Informatika		Nilai :