4. Nesneleri Hareket Ettirme

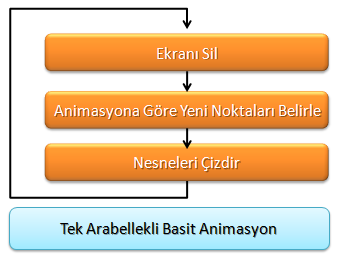
Bu başlıkta öğreneceğiniz konular

* Nesneleri hareket ettirme
* Tek arabellekli hareket
* Çift arabellekli hareket (Önerilen)

## Tek Arabellekli hareket

###### Basit hareket ettirme metodu

Arkadaşlar nesneleri hareket ettirmenin bir animasyon hazırlamanın en basit yöntemi ekranı tamamıyla silmek(1), nesnelerin yeni konumunu belirlemek(2) ve nesneleri çizdirmektir(3).



Şimdi örnek kodlarla bu algoritmayı inceleyelim.

#### **#include <graphics.h>**

#### **int yon=1;**

#### **int konumx=50;**

#### **void yenikonum(){**

#### **if (konumx<351 && yon==1) konumx+=5;**

#### **if (konumx==350) yon=0;**

#### **if (konumx>49 && yon==0) konumx-=5;**

#### **if (konumx==50) yon=1;**

#### **}**

#### **int main()**

#### **{**

#### **initwindow(400,300, "3. Nesneleri Hareket Ettirme");**

#### **while(1){**

#### **cleardevice();**

#### **yenikonum();**

#### **setcolor(255);**

#### **circle(konumx,50,20);**

#### **setcolor(COLOR(0,255,0));**

#### **circle(50,150,20);**

#### **delay(40);**

#### **}**

#### **closegraph();**

#### **return 0;**

#### **}**

yenikonum(); kısaca x=50 ile x=350 arasında gidip gelen bir noktanın yeni konumunu belirlemek için kullandık.

Kodlardaki animasyon algoritması şu kodlardadır.

#### **while(1){**

#### **cleardevice();**

#### **yenikonum();**

#### **setcolor(255);**

#### **circle(konumx,50,20);**

#### **setcolor(COLOR(0,255,0));**

#### **circle(50,150,20);**

#### **delay(40);**

#### **}**

Döngü içerisindeki kodları açıklayalım.

# cleardevice ( );

cleardevice(); Ekranı arkaplan rengine göre tamamen temizler.

# circle ( x, y , r [yarıçap] );

circle(x,y,r); Merkezi x,y noktası olan yarıçapı r olan bir daire çizer.

# delay ( süre [milisaniye] ) ;

delay(s); s milisaniye kadar programı bekletir. delay(1000); 1000 milisaniye = 1 saniye kadar programı bekletir.

Filmler olsun oyunlar olsun genelde saniyede 25 kare fotoğrafın ardı ardına gelmesiyle gözümüzde canlılık kazanır.

Bu da yaklaşık her kareden sonra 1000 ms / 25 kare = 40 ms/kare beklemek gerekir. Bu işlemide delay(40); ile yapıyoruz.

Görüldüğü üzere önce ekranı temizledik, animasyon konumunu belirledik, nesneyi çizdirdik, ve sonraki animasyon için biraz bekledik.

Tek Arabellekli Hızlı Animasyon



Tek arabellekli Basit Animasyonda görüntünün tamamını siliyorduk. Fakat şimdi ekranın tamamını değilde animasyonla değişicek yerleri saptayıp eski yerleri temizleyip(1), animasyondaki yeni yerleri saptayıp(2), Nesneleri çizdiriyoruz(3).

Örnek

#### #include <graphics.h>

#### int yon=1;

#### int konumx=50;

#### void yenikonum(){

#### if (konumx<351 && yon==1) konumx+=5;

#### if (konumx==350) yon=0;

#### if (konumx>49 && yon==0) konumx-=5;

#### if (konumx==50) yon=1;

#### }

#### int main()

#### {

#### initwindow(400,300, "3. Nesneleri Hareket Ettirme");

#### setcolor(255);

#### while(1){

#### setcolor(0);

#### circle(konumx,50,20);

#### yenikonum();

#### setcolor(255);

#### circle(konumx,50,20);

#### setcolor(COLOR(0,255,0));

#### circle(50,150,20);

#### delay(40);

#### }

#### closegraph();

#### return 0;

#### }

Bir önceki örneğin aynısı fakat değişen kodlar şöyle;

#### while(1){

#### setcolor(0);

#### circle(konumx,50,20);

#### yenikonum();

#### setcolor(255);

#### circle(konumx,50,20);

#### setcolor(COLOR(0,255,0));

#### circle(50,150,20);

#### delay(40);

#### }

cleardevice(); yi kaldırdık. Yerine eski nokları silmek için setcolor(0); ve circle(konumx,50,20); bu sayede ekranın tamamını temizlemek yerine sadece animasyon yapılan bölgeyi temizledik.(1)

Ardından yenikonum(); ile animasyonumuzdaki yeni konumları belirledik.(2)

Son olarak da setcolor(255); ve circle(konumx,50,20) ile yeni animasyonumuzu çizdirdik.(3)

delay(40); bizim animasyonda olmazsa olmazımızdı zaten.

Kısaca konuyu şöyle özetleyebiliriz;



Çift Arabellekli Animasyon



# void swapbuffers ( );

Çift Arabellekli Animasyon oyun sanayisinin olmazsa olmazıdır. Günümüzdeki oyunların bir çoğu bu teknikle animasyon yapar. Çift Arabellekli Animasyonu anlatmak gerekirse; çizim işlemlerini kullanıcıya farkettirmeden arka bellekte yapar görüntülemeyi ön bellekte yapar. Çizim işlemleri bittikten sonra yeni kare arka bellekten, ön belleğe swapbuffers(); kodu ile geçiş yapar. Bu sayede kullanıcı her zaman çizimleri tamamen bitmiş görüntüleri güçlü bir şekilde gözü yorulmadan görür.

Örnek :

#### #include <graphics.h>

#### int yon=1;

#### int konumx=50;

#### void yenikonum(){

#### if (konumx<351 && yon==1) konumx+=5;

#### if (konumx==350) yon=0;

#### if (konumx>49 && yon==0) konumx-=5;

#### if (konumx==50) yon=1;

#### }

#### int main()

#### {

#### initwindow(400,300, "3. Nesneleri Hareket Ettirme");

#### swapbuffers();

#### while(1){

#### cleardevice();

#### yenikonum();

#### setcolor(255);

#### circle(konumx,50,20);

#### setcolor(COLOR(0,255,0));

#### circle(50,150,20);

#### swapbuffers();

#### delay(40);

#### }

#### closegraph();

#### return 0;

#### }

Ek olarak sadece swapbuffers(); ’ i ekledik. Daha öncesinde dediğimiz gibi arka bellekteki görüntüyü ön bellek ile değiştirir.

swapbuffers(); ’ı tek kullanımlık olarak initwindow(); fonksiyonundan sonra kullanmanızı öneririz. Çünkü ilk çizimdelerde uygulama Çift Arabellekli Animasyon Kullandığımızı bilmiyecektir ve bu yüzden ilk çizim animasyonunu Tek Arabellekli Animasyon olmasını istemediğimiz için eklemeliyiz.

swapbuffers(); fonksiyonunu bir kez dahi olsa kullandığınız anda program Çift Arabellekli Çizim Metoduna geçtiği için görüntünün yeninlenmesini istediğiniz zamanlarda swapbuffers(); kullanmanız zorunludur. Yoksa uygulamanızda yeni görüntüler oluşmaz.

Bundan sonraki bütün uygulamalarımızda kullanıcağımız Çift Arabellekli Animasyon’ un özetlemesi şu şekildedir.

