Karadeniz Teknik Üniversitesi

Programlama Dili Kavramları

Final Projesi

385961 Erdem Akgün

385960 Batırhan Berk Fil

385954 Mert Akın

Ocak 2022

Amaç

Projemizin amacı 7'den 70'e insanların sıkıldıklarında biraz olsun sorumluluklardan uzaklaşıp kafalarını dağıtmak için eğlenceli vakit geçirebilecekleri bir oyun.

Oyunumuzda bulunan gemimiz ile uzay boşluğunda hareket edip gelen asteroitlerden kaçarak ve onları patlatarak ilerlemek. Asteroitlerden çıkan güçlendirme özellikleri ve asteroit patlattıkça biriktirdiğimiz ulti ile yolumuza devam etmek. Seviyeleri geçtikçe zorlaşan oyunumuzda amacımız gidebileceğimiz kadar ileri gitmek.

Oyunumuzda yukarı, aşağı yön tuşları ile gemimizi hareket ettiriyoruz. Boşluk tuşu ile ateş ve x tuşu ile ulti özelliğimizi kullanabilmekteyiz. Oyunumuzu başlatmak için ise "p" tuşuna basılması gerekiyor.

Proje İçeriği

1-) Resim, ses ve kütüphanelerin projeye entegre edilmesi

2-) Temel komutların ayarlanması

3-) Sınıfların oluşturulması

4-) While döngüsü kurulması ve kodların çalıştırılması

1-) Resim, ses ve kütüphanelerin projeye entegre edilmesi

Öncelikle pygame, os.path, random, sys sınıflarımızı projeye import edilir.

```
import pygame
```

os.path.dirname() fonksiyonumuz ile klasörümüzde bulunan oyunumuz için gerekli olan resim ve müziklerimizi eklemek için klasör tanımlanır.

```
resimKlasoru = os.path.join(klasor, "resimler")
background = pygame.image.load(os.path.join(resimKlasoru, "farback.gif"))
```

Komutumuz ile diğer resim dosyalarımızı da örnekte olduğu gibi değişkenlere atanır.

Ses dosyalarımız içinde benzer şekilde işlemler uygulanır.

```
sesKlasoru = os.path.join(klasor, "sesler")
fireEffect = pygame.mixer.Sound(os.path.join(sesKlasoru, "fire.ogg"))
```

2-) Temel komutların ayarlanması

Temel ayarlarımızda penceremizin boyutunu ayarlanması için genişlik yükseklik verileri değişkenlere atanır. Penceremizi oluşturmak ve penceremize oyunumuzun resmini ve ismini koymak için gerekli komutlar yazılır.

```
pencere = pygame.display.set_mode(boyut)
pygame.display.set_icon(ship)
pygame.display.set_caption('Space Shooter')

pygame.mouse.set_visible(False)
font = pygame.font.SysFont("Helvetica", 50)

clock = pygame.time.Clock()
```

Oyunumuzda farenin işlevi olmadığı için fare devre dışı bırakılır. Oyunumuzda kullanacağımız yazı fontlarını ayarlamak için font değişkeni ayarlanır. Pygame.time.Clock() fonksiyonundan clock adında zamanı izlememizi sağlayan bir nesne oluşturulur.

3-) Sınıfların Oluşturulması

Sınıflarımızı yeni py dosyalarında oluşturup ana sınıfımıza import ediyoruz.

Parca

Oyunumuz uzay oyunu olduğu için bir uzay gemisi nesnesi oluşturmamız gerekiyor. Bunun için Parca isminde bir sınıf oluşturuyoruz. Sınıfımızda ilk olarak yapıcı metodumuzu yazıyoruz. Sınıfımıza pygame.sprite.Sprite sınıfından miras alıyoruz ve bu sınıfın fonksiyonlarını kullanmak bilmek için super().__init__() fonksiyonumuzu yazıyoruz.

Şimdi ise parçamızın değerlerini atıyoruz.

Resmi için image değişkeni oluşturup gemi resmimizi daha hızlı çizilmesi için convert ediyoruz. Can değerimizi atıyoruz. Arka plan rengimizi ayarlıyoruz. Resmimizin hareket etmesi için rect sınıfımızı çağırıyoruz. Dikdörtgen spirtemızı 2d de hareket ettirebilmemiz için bu işlemi yapıyoruz. Rect.x ve rect.y değerlerini atıyoruz ardından. Gemimiz için kalkan ve mermi hızı değerlerinin de atamalarını ekliyoruz.

Mermi hızını değiştir adında bir fonksiyon oluşturuyoruz. Bu fonksiyon sayesinde mermi hızımı alacağımız güçlendirme ya da güçsüzleştirme etkileriyle düşürme ya da hızlandırma yapacağız.

Gizlenme adında bir fonksiyon oluşturuyoruz. Bu fonksiyonumuz gemimizin bir can kaybettiğinde pencerede verdiğimiz boyutların dışına çıkartıyoruz. Böylelikle kullanıcıdan gizlenmiş oluyor. Bu süreyi 1,5 saniye olarak ayarlıyoruz.

Update fonksiyonumuz ile sürekli parçamızı güncelliyoruz. Bunun sebebi pythonda aslında resimleri oynatmamız. Resimleri hızlı oynattığımız için bize video şeklinde geliyor.

*args komutu ile sınırsız parametre gönderebiliyoruz. Bu sayede tuş basmalarımızı if koşullarıyla kontrol edip parçamızı hareket ettirebiliyoruz. Gizlenme ayarlarımızı yapabiliyoruz. Belirli seviyelere geldiğimizde gemimizin rengini değiştirebiliyoruz.

Güçlendirmelerimizin ayarlarını kontrol edebilmek için pygame.time.get_ticks() komutundan yararlarınıyoruz. Bu komut oyun başladığı andan itibaren geçen süreyi veriyor.

Shoot fonksiyonumuz da ise gemimizin ateş etmesini sağlıyoruz. Burada müzik efekti ve ateşin gemimizin hangi konumundan ateş edeceğini ayarlıyoruz.

Ardından bunu sprites gruplarımıza ekliyoruz. Seviyemiz belirli bir yere gelince 2. Bir ateş daha ekliyoruz. Bunu da if koşuluyla sağlıyoruz.

Mermi

Oyunumuz için asteroitler oluşturmamız gerek. Gemimizin bunları vurarak ilerlemesi için. Bunun için Mermi adında bir sınıf oluşturuyoruz. pygame.sprite.Sprite sınıfından miras alıyoruz ve super fonksiyonu ile çağırıyoruz. Asteroitlerimizin hepsinin boyutu farklı olduğu için bunları random seçmek istiyoruz bu yüzden random.choice(asterodiler) komutunu kullanarak farklı boyutlarda resimler yüklediğimiz asteroitler dizimizden seçim yapıyoruz. Ardından convert ve arka plan işlemlerini ayarlıyoruz. Asteroitlerimizin gemimize değmesinde çarpma olayını ayarlamak için asteroitlerimizin boyutunun %70'ini alıyoruz.

Penceremizde nereden çıkacağını ve hızlarını random bir şekilde belirlemesini istiyoruz ve hareket ederlerken kendi etraflarında dönmelerini istiyoruz. Bunun için döndürme isimli bir fonksiyon oluşturuyoruz. Burada mermi asteroidimizin kendi çevresinde dönmesini ve şeklini korumasını sağlıyoruz.

Update fonksiyonumuz ile pencere dışına çıkan asteroitlerimizin tekrar oluşturulurken hızlarının ve çıkış konumlarının tekrardan random ataması işlemini ayarlıyoruz ve vurulan asteroit karşılığında skorumuzu 1 puan artırıyoruz.

Fuze

Asteroitleri vurmamız için gemimize ateş ettirecek sınıf olan Fuze sınıfını oluşturuyoruz. Lazerimizin resim dosyasını ekliyoruz. Füzenin gemimizin tam ortasından çıkması için koordinat ayarlarını ekliyoruz. Update fonksiyonu ile x koordinatında gideceği hızı ve if koşulu ile genişliğimizin dışına çıktığında yok sağlıyoruz.

Patlama

Bu sınıfımızda asteroitleri vurduğumuzda oluşmasını istediğimiz patlama efektini çıkartacağız. Bunun için patlama klasöründe resimleri for döngüsü yardımıyla patlamaResimleri isimli değişkenimize append ediyoruz.

Fonksiyonumuz da resimleri değişkenlerimize atama işlemleri yapıyoruz. Resimler arasında olacak süreyi delay olarak belirtiyoruz. Update fonksiyonumuzda ise geçen süre ve o an tuttuğumuz süreyi birbirinden

çıkartarak delay süresinden büyük olup olmadığını sorguluyoruz. Eğer öyleyse resimlerimiz tek tek karşımıza çıkıyor ve resim döngüsü bittiğinde if koşuluyla bunu sonlandırıyoruz.

Canciz

Can ciz sınıfımızda mevcut bulunan canlarımızı penceremizde göstermek istiyoruz. Bunun için gemimizin resmini alıyoruz ve boyutunu küçültüyoruz. Resmimizin çıkacağı kısmın koordinatlarını ayarlıyoruz. Ardından penceremize ekliyoruz.

KalkanCiz

Kalkan değerimiz için penceremizde bir bar oluşturduk. Barın yükseklik ve genişlik ayarlarını atıyoruz. Barımızı gelen değer değişkenine göre if koşulları ile rengini belirliyoruz. Değer değişkeni düştükçe rengimizde değişiyor.

Ulti

Ulti sınıfımızda kalkan ciz sınıfımıza benzer kodlarımızı yazarak ulti barı oluşturuyoruz. Bu bar skordan aldığımız değerle yükseliyor ve rengi belirleniyor.

PowerUp

Bu sınıfımızda asteroitlerimizi vurduğumuzda %10 şans ile düşecek güçlendirmeleri eklemek istiyoruz. Bunun için resimlerimizi powerUps adlı değişkenimize atıyoruz. PowerUp isimli fonksiyonumuzda rastgele bir değişken seçmesini ardından resmin x ekseni boyunca ilerlemesini sağlıyoruz.

GameOver

Oyunumuzu bitiş ve başlangıcını bir döngüde yapmak için GameOver sınıfını açıyoruz. Show_gameover_screen fonksiyonumuz ile bir başlangıç resmi ekliyoruz. Ardından açtığımız while döngüsü ile çıkış ve başlatma durumlarını kontrol ediyoruz. Kullanıcı başlatmak isterse p tuşuna basarak oyunu tekrar başlatabiliyor.

4-) While döngüsü kurulması ve kodların çalıştırılması

Oyunumuz hareket eden resimlerden oluştuğu için sürekli tekrar etmesi gerek. Bunun için bir while döngüsü açıyoruz. Oyunun gameover durumunu sorgulamak için bir if koşulu açıyoruz. Burada oyun müziklerimizi ve başlangıç ekranını veriyoruz. Gruplarımızı ekliyoruz. İlk asteroitlerimizi oluşturuyoruz. Uzay gemimizi oluşturuyoruz. Oyundaki asteroit sayımızı bir değişkene atıyoruz ve bunu ekrana yazdırıyoruz. Penceremizin arka plan rengini ayarlıyoruz. İf koşulları ile level atladıkça arka plan resimleri değişiyor. Penceremiz draw ile ekrana çizdirildi.

Pygame .QUIT: sys.exit() komutumuz ile penceremizde x(çarpı)'e basıldığında kapatmayı ayarlıyoruz. Keys parametreleri ile hangi tuşa basıldığında hangi işlemlerin yapılacağını belirtiyoruz.

Pygame.sprite.groupcollide komutu ile 2 sprite mızın çarpışmasını denetliyoruz. Bunlar bizim füzelerimiz ve asteroitlerimiz. Eğer bu koşul gerçekleşirse skorumuz 1 artar. Ulti değerimiz 1 artar. Vuruş efektimiz çalıyor. Patlama efektlerimiz oluşuyor. %10 olasılıkla powerups düşme şartımızı gerçekleştiriyoruz ve yeşil asteroitler vurursa eğer kalkan değerimiz 10 puan artar.

```
isPowerGain = pygame.sprite.spritecollide(parca1, powerGains, True)
```

Bu sefer gemimiz ile güçlendirmelerimizin çarpışmasını denetliyoruz. Eğer bir güçlendirme alırsa karakterimiz if koşulunun içine girer ve aldığı güçlendirmeye bağlı if koşullarına girerek efekt müziği ve kazandığı güçlendirmeyi aktif hale getirir.

```
durum = pygame.sprite.spritecollide(parca1, mermiler, True,
collided=pygame.sprite.collide_circle)
```

Bu çarpışmada gemimiz ile asteroitlerin çarpışma durumu ayarlıyoruz. Asteroit gemimize vurduğunda çarpma efektini çalıyor. Ardından patlama resimlerimiz geliyor ve çarpan asteroittin yarıçapının 3 katı hasarı kalkanımıza verir.

```
Kalkan.kalkanCiz(pencere, 5, 5, parca1.kalkan)
CanCiz.canCiz(pencere, 5, 25, parca1.can)
Ulti.ultiCiz(pencere, 5, 50, Ulti.ulti)
```

Burada kalkan, can ve ulti fonksiyonlarını çalıştırıyor.

```
if Ulti.ulti == 100:
```

Eğer ulti durumumuz 100'e eşitse x tuşuna basıldığında aktif hale gelir. Ulti değeri 0 olur ve mermi hızı aşırı şekilde artar.

```
if durum or mermiSayisi == 0:
```

Mermi sayımız 0'a eşitse bu koşula girer. Burada kalkanımız 0'a eşitse veya daha düşük bir değerde ise patlama efektimiz çalar. Ardından can değerimiz 1 azalıyor ve gemimiz saklanma fonksiyonunu çağırır. Can değerimiz 1 azaldığında 0 değerine ulaşıyorsa game_over durmumuz True olur ve oyun sonlanır. Değilse kalkan değerimiz 100 olarak tekrar oyuna devam eder.

Asteroit durumumuzun 0 olduğu diğer bir if koşulunda ise level durumumuzu 1 arttırıp ekrana yazdırıyoruz ve diğer level in gelmesini 4 saniye bekliyoruz. Diğer level için for döngümüzde level*10 şeklinde asteroit üretilir.

Son olarakta

```
pygame.display.update()
```

komutumuz ile bellekte oluşturduğumuz bu komutlar ekrana çağrılır.