CHRONA PROJECT: TEMPORAL FALLOUT

YouTube Linki:

https://youtu.be/XSUBDjlQ0T8?si=DjY3temHf7b2pvWY

İndirme Linki:

https://drive.google.com/file/d/1eFFWn9ej0UJuFRIkKmYbdMR0T1z1Uqp_/view?usp=sharing

Oyun Türü: 2D Platform – Bilim Kurgu, Macera, Aksiyon
Hedef Platform: PC
Geliştirme Motoru: Unity

Kısa Tanım:
Chrona Project: Temporal Fallout, nükleer füzyon temelli bir zaman enerjisi deneyinin başarısız olmasından sonra, iki farklı zaman diliminde (2054 ve 2254) geçen bir 2D platform oyunudur. Oyuncu, bilim insanı Elias Korrin'i kontrol eder. Elias, insanlığın kaderini değiştiren bir hatayı düzeltmek için geçmişin yeşil dünyası ve geleceğin radyasyonla bozulmuş yıkıntıları arasında geçiş yaparak "Chrona Reaktörü"nün dengesini yeniden sağlamaya çalışır.
İnsanlık, enerji krizini çözmek için nükleer füzyonun gücünü zaman fiziğiyle birleştiren **Chrona Project** adında bir proje başlatır.
Projenin amacı, füzyon sırasında ortaya çıkan yüksek enerjili nötronların, zamanın kuantum alanında oluşturduğu dalgalanmalardan enerji elde etmektir.

Projenin başındaki bilim insanı Dr. Elias Korrin, "Chrona Partikülleri" adını verdiği bu yeni enerji formunu kontrol altına almayı başarır. Ancak ilk büyük deney sırasında reaktör çekirdeği aşırı yüklenir ve zaman alanı çöker. Bu olay tarihe Zaman Çökmesi (Temporal Collapse) olarak geçer.

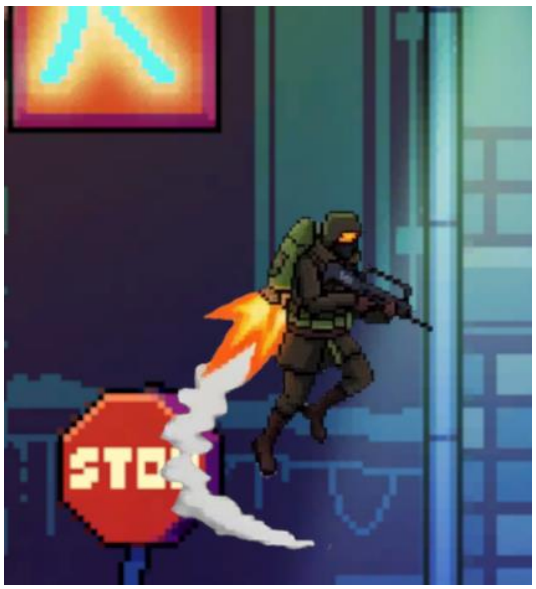
1. OYUNCU KONTROL MEKANİKLERİ (PLAYER CONTROLLER)

Oyuncu sistemi, hareket ve savaş mekaniklerini birbirinden bağımsız ancak koordineli çalışan scriptler üzerinden yönetmektedir. Bu yapı, kodun okunabilirliğini ve yönetilebilirliğini artırmaktadır.

1.1. Fizik Tabanlı Hareket ve Çift Zıplama

Oyuncu hareketi, Unity'nin fizik motoru (`Rigidbody2D`) üzerine inşa edilmiştir. `PlayerMovement` sınıfı, oyuncunun yatay eksendeki hızını (velocity) doğrudan manipüle ederek hassas bir kontrol sağlar.

- **Double Jump (Çift Zıplama):** Oyuncunun havada bir kez daha zıplamasına olanak tanıyan sayaç tabanlı (`jumpCount`) bir sistem entegre edilmiştir.
- **Zemin Kontrolü (Ground Check):** `Raycast` yöntemi kullanılarak oyuncunun zemine teması anlık olarak kontrol edilir, bu sayede havada sonsuz zıplama gibi hataların önüne geçilir.



Şekil 1: Player zıplama görseli

1.2. Gelişmiş Çömelme (Crouch) Sistemi

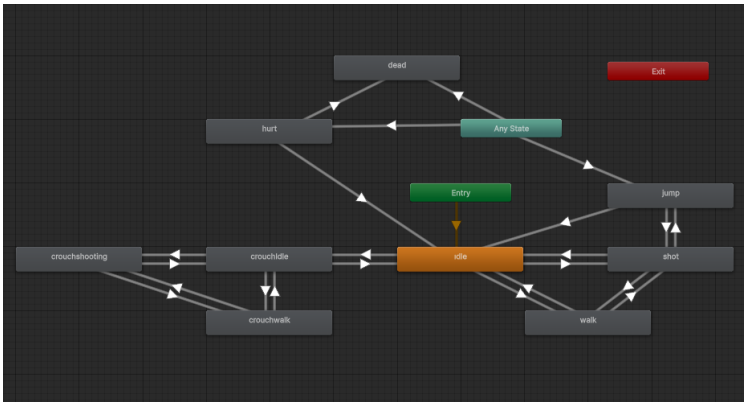
Standart oyunların aksine, çömelme mekaniği sadece karakterin fiziksel boyutunu (Collider) küçültmekle kalmaz, aynı zamanda karakteri farklı bir animasyon katmanına geçirir. `CrouchMovement` sınıfı, oyuncunun çömelerek yürümesine ("CrouchWalk") ve bu pozisyondayken etkileşime girmesine olanak tanır.

Şekil 2: Player eğilme ve ayakta ateş etme görselleri

1.3. Savaş Sistemi ve Atış Mekaniği

Savaş sistemi, hem ayakta hem de çömelme pozisyonunda ateş etmeyi destekler (`PlayerShooting` & `CrouchShooting`).

- **Dinamik Ateş Noktaları:** Mermilerin çıkış noktası, karakterin duruşuna (ayakta veya eğilmiş) göre dinamik olarak değişir.
- **Animasyon Senkronizasyonu:** Ateş etme eylemi sırasında hareket kısıtlamaları uygulanarak animasyonun kayması önlenir.



Şekil 3: Oyuncu Sonlu Durum Makinesi (State Machine) Yapısı. "Any State" geçişleri sayesinde hasar alma durumu her an tetiklenebilir.

2. YAPAY ZEKA (AI) VE DÜŞMAN SİSTEMLERİ

Düşmanlar, durum tabanlı (State-Based) bir yapay zeka mimarisi ile yönetilmektedir. Sistem; devriye, tespit etme, kovalama ve saldırı fazlarından oluşur.

2.1. DOTween ile Devriye (Patrol) Sistemi

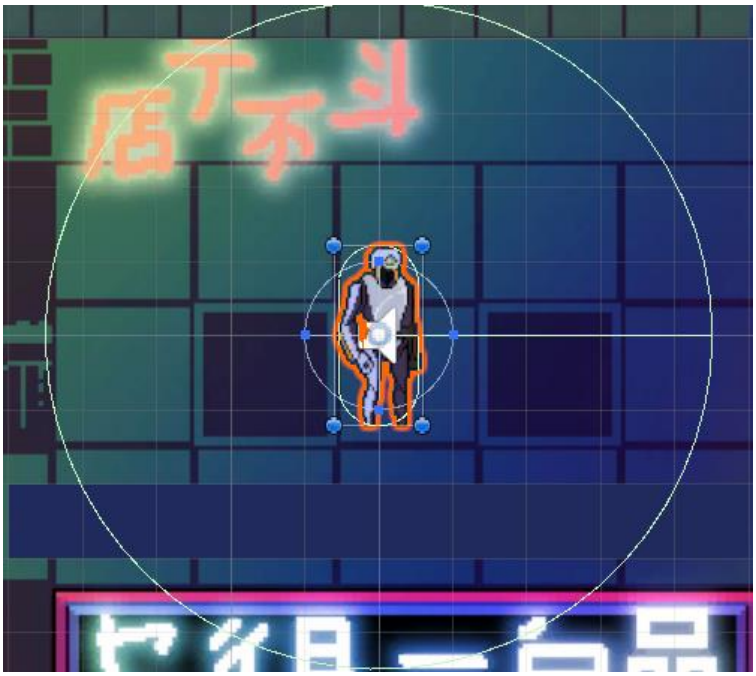
Düşmanların devriye hareketleri için **DOTween** kütüphanesi kullanılmıştır. Bu sistem, düşmanın A noktasından B noktasına "Sequence" (Sıralı İşlem) yapısıyla pürüzsüzce hareket etmesini sağlar.

- **Özellik:** `EnemyPatrolTween` scripti, düşmanı belirlenen noktalara götürürken fizik motorunu (Kinematic Body) yormadan interpolasyon (Tweening) tekniklerini kullanır. Düşman bir engelle karşılaştığında veya oyuncuyu gördüğünde bu "Tween" hareketi anında durdurulabilir.

2.2. Tespit ve Takip (Detection & Chase)

Yapay zeka, hibrit bir algılama sistemine sahiptir:

1. **Trigger Alanları:** `EnemyDetectionTrigger` scripti, oyuncunun belirli bir menzile girdiğini fiziksel çarpışma (Collider) ile algılar.
2. **Davranış Değişimi:** Oyuncu tespit edildiğinde, devriye modu (`Kinematic`) devre dışı bırakılır ve fizik tabanlı takip modu (`Dynamic Rigidbody`) devreye girer. Bu, düşmanın yer çekiminden ve çevresel fizikten etkilenerek oyuncuyu kovalamasını sağlar.

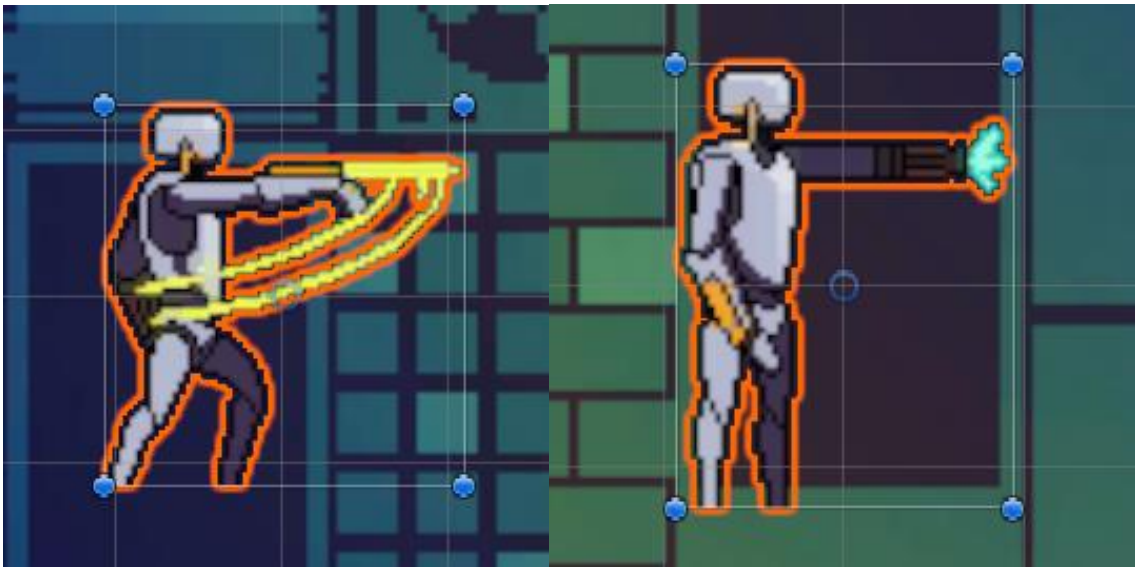


Şekil 4: Enemy Trigger Alanı

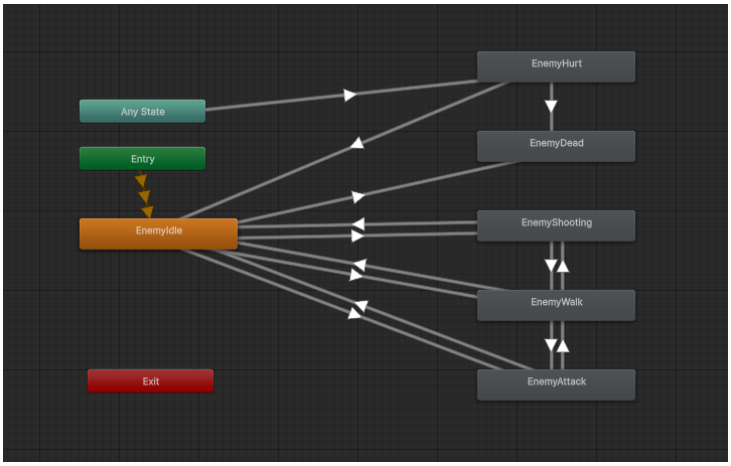
2.3. Saldırı Varyasyonları

Sistem hem yakın dövüş (Melee) hem de menzilli (Ranged) düşman tiplerini destekler:

- **Melee AI:** Oyuncuya belirli bir mesafeye kadar yaklaşır, durur ve saldırı animasyonunu tetikler.
- **Ranged AI:** Oyuncuyu gördüğü anda durur, oyuncunun yönüne döner ve belirli aralıklarla mermi (Projectile) instantiate eder.



Şekil 5: Düşman Tipi ve Saldırı Türleri



Şekil 6: Düşman Yapay Zekası Animasyon Akışı. Saldırı, Yürüme ve Ölüm durumları arasındaki geçişler.

3. MERKEZİ SAĞLIK VE HASAR SİSTEMİ

Projede "Tek Sorumluluk Prensibi" (Single Responsibility Principle) gözetilerek `Health` sınıfı geliştirilmiştir. Bu sınıf hem oyuncu hem de düşmanlar için ortaktır.

- **Hasar Alma (Take Damage):** Can değeri düştüğünde UI (Health Bar) otomatik güncellenir.
- **Stun (Sersemletme) Mekaniği:** Bir düşman hasar aldığında, `Health` scripti düşmanın hareketini ve saldırısını geçici olarak durdurur (`Hurt` animasyonu). Bu, vuruş hissini (Game Feel) güçlendiren kritik bir detaydır.
- **Ölüm Yönetimi:** Can sıfıra indiğinde ilgili nesnenin collider'ları kapatılır, ölüm animasyonu oynatılır ve nesne sahneden temizlenir.

---

4. OYUN YÖNETİMİ VE SİSTEM MİMARİSİ

Oyunun akışı, sahneler arası veri taşıma ve kullanıcı arayüzü yönetimi "Manager" sınıfları ile kontrol edilmektedir.

4.1. Singleton ve Persistent Data

`PersistentManager` sınıfı, **Singleton** tasarım deseni ile oluşturulmuştur. Bu obje `DontDestroyOnLoad` ile işaretlenerek sahneler arası geçişte yok olmaz. Böylece oyunun genel durumu korunur.

4.2. Kamera Sistemi (Cinemachine)

Kamera takibi için Unity'nin **Cinemachine** paketi kullanılmıştır.

- **Level Yönetimi:** `LevelManager` sınıfı, oyuncu sahneler arası ışınlandığında (Teleport) kameranın hedefi kaybetmesini veya ani sıçramalar yapmasını önlemek için `OnTargetObjectWarped` metodunu kullanarak kamerayı anında yeni konuma odaklar.

4.3. Kullanıcı Arayüzü (UI) ve Menüler

- **Video Entegrasyonu:** Ana menüde `VideoPlayer` bileşeni kullanılarak dinamik (patlama efektli) bir arka plan oluşturulmuştur.
- **Ayarlar ve Veri Kaydı:** `VolumeSettings` sınıfı, ses ayarlarını yönetir ve `PlayerPrefs` kullanarak kullanıcının ses tercihini cihaz hafızasına kaydeder.
- **Pause Sistemi:** Oyun durdurulduğunda `Time.timeScale = 0` yapılarak tüm fizik ve animasyon zamanlaması dondurulur.

Şekil 7: Ana Menü

Şekil 8: Duraklatma Menüsü

5. SONUÇ

Bu proje, modern oyun geliştirme standartlarına uygun, genişletilebilir bir altyapıya sahiptir. State Machine (Durum Makinesi) yapısının animatör ile entegre kullanılması, DOTween ile programatik animasyonların yönetilmesi ve modüler script yapısı; projenin ilerleyen aşamalarında yeni düşman tiplerinin, bölümlerin veya mekaniklerin sisteme kolayca eklenebilmesine olanak tanımaktadır.

Şekil 9: Oyun içi görsel

CREDİTS

- https://ludo.ai/
- https://gemini.google.com/?hl=tr
- https://hailuoai.video/
- https://freesound.org/
- https://craftpix.net/?srsltid=AfmBOoqX3DTY4Nb_RF2145fvUgh-j3vwQsErdYJ5ruX7kCe_Jzk2VriQ
- https://itch.io/