Unity Defter [Tips](#Tips)

Anahtarlar

[\*A => Alfa](#Alfa)

[\*B => Beta](#Beta)

[\*G => Gama](#Gama)

Siteler :: dofont.com

textures.com

freesound.org

freesound.com

blender

sketchfab.com

Asset siteleri

* + sketchfab.com
  + miximo.com
  + free3d.com

[(Sorun-Silme-Package Manager)](#Sorun_Silme_Package_Manager)

[(Switch-Case’de ||)](#Switch_Case_veya)

[(Temel Fonksiyonlar)](#Temel_Fonksiyonlar)

[(Fixed Update)](#Fixed_Update)

[(Coroutine’yi Privateye Ata)](#Coroutineyi_Privateye_Ata)

[(Soru-1)](#Soru_1)

[(Vektörler ve Equals)](#Vector_Equals)

[(Soru-2)](#Soru_2)

[(Hide Flags)](#Hide_Flags)

[(Soru-3)](#Soru_3)

[(Soru-4)](#Soru_4)

[(Soru-5)](#Soru_5)

[(Soru-6)](#Soru_6)

[(Soru-7)](#Soru_7)

[(Soru-8)](#Soru_8)

İçindekiler Tablosu

[Bölüm-1: Genel Bilgilendirme 13](#_Toc145072786)

[1-Tanıtım 13](#_Toc145072787)

[2-Örnek Oyun Kazanç Tablosu 13](#_Toc145072788)

[3-3D-2D Oyun Nedir? Aralarındaki Fark Nedir? 14](#_Toc145072789)

[4-Oyun Türleri Nelerdir? 14](#_Toc145072790)

[5-Merak Edilen Sorular ve Cevapları 14](#_Toc145072791)

[Bölüm-2: Unity Kurulum ve Tanıtım 14](#_Toc145072792)

[1-Unity Hub Kurulum 14](#_Toc145072793)

[2-Unity Arayüz Tanıtımı ve Asset Store 14](#_Toc145072794)

[Bölüm-3: Unity-Temel C# Dersleri 15](#_Toc145072795)

[1-Değişken Tanımlama Türleri 15](#_Toc145072796)

[2-If-Else 15](#_Toc145072797)

[3-For Döngüsü 15](#_Toc145072798)

[4-Array 15](#_Toc145072799)

[5-Foreach 15](#_Toc145072800)

[6-While-Do Döngüsü 15](#_Toc145072801)

[7-Switch-Case 15](#_Toc145072802)

[8-List Kavramı 15](#_Toc145072803)

[9-Tuple Patterns(İleri Seviye) 15](#_Toc145072804)

[10-Property Patterns(İleri Seviye) 15](#_Toc145072805)

[11-Properties Kullanımı(İleri Seviye) 15](#_Toc145072806)

[Bölüm-4: Unity Arayüz Tanıtımları ve Çeşitli İşlemler 15](#_Toc145072807)

[1-Sürüm ve Platform Yönetimi 15](#_Toc145072808)

[2-Console Penceresi Anlatımı-Debug Çeşitleri 15](#_Toc145072809)

[3-Scene ve Game Pencereleri 16](#_Toc145072810)

[Bölüm-5: Unity Fonksiyonları ve Kodları 16](#_Toc145072811)

[1-Class-Fonksiyon-I-II-III-IV 16](#_Toc145072812)

[6-Using Kullanımı-Namespace 16](#_Toc145072813)

[7-MonoBehaviour-Messages 16](#_Toc145072814)

[8-MonoBehaviour-Coroutine 16](#_Toc145072815)

[9-MonoBehaviour-Invoke 17](#_Toc145072816)

[Bölüm-6: Unity’ye İlk Adımlar-Çeşitli Obje İşlemleri-Isınma Turları 17](#_Toc145072817)

[1- Objelerin Anlatımı-Scale-Move-Rotate ve Diğer Detaylar 17](#_Toc145072818)

[2-Inspector Detayları ve Component Anlatımları-I 17](#_Toc145072819)

[3-Inspector Detayları ve Component Anlatımları-II 17](#_Toc145072820)

[4-Inspector Detayları ve Component Anlatımları-III 17](#_Toc145072821)

[5-Boş Obje Oluşturma ve Görevleri-Hierarchy Child 17](#_Toc145072822)

[6-Objeye Değişken Tanımlama ve Çeşitli İşlemler 17](#_Toc145072823)

[7-Array-Diziler ve List Anlatım ve Tanımlama 18](#_Toc145072824)

[8-GetComponent Anlatımları ve İşlemleri-I 18](#_Toc145072825)

[9-GetComponent Anlatımları ve İşlemleri-II 18](#_Toc145072826)

[10-Obje Yakalama-GameObject.Find-FindWithTag-FindGameObjectsWithTag-transform.Find-I 18](#_Toc145072827)

[11-Obje Yakalama-GameObject.Find-FindWithTag-FindGameObjectsWithTag-transform.Find-II 18](#_Toc145072828)

[12-GetComponent typeof-FindObjectOfType-SetActive-activeSelf 18](#_Toc145072829)

[13-CompareTag-AddComponent 18](#_Toc145072830)

[14-BroadCastMessage-SendMessage 18](#_Toc145072831)

[15-Gameobject Destroy-Component Destroy 18](#_Toc145072832)

[Bölüm-7: Unity Transform&Vector&Quaternion İşlemler ve Kodlar-Obje Oluşturma 19](#_Toc145072833)

[1-Proje-Import Package-Export Package 19](#_Toc145072834)

[2-Prefab Oluşturma ve Anlatımı 19](#_Toc145072835)

[3-Transform Translate-Vector3 Static Özellikler-I 19](#_Toc145072836)

[4-Transform Translate-Vector3 Static Özellikler-II 19](#_Toc145072837)

[5-Vector3-Özellikler ve Public Metotlar 19](#_Toc145072838)

[6-Vector3-Static Methodlar-I 19](#_Toc145072839)

[7-Vector3-Static Methodlar-II 19](#_Toc145072840)

[8-Vector3-Static Methodlar-III ve Operatörler 19](#_Toc145072841)

[9-Transform-Özellikler ve Methodlar-I 19](#_Toc145072842)

[10-Transform-Özellikler ve Methodlar-II 20](#_Toc145072843)

[11-Transform-Özellikler ve Methodlar-III 20](#_Toc145072844)

[12-Transform-Özellikler ve Methodlar-IV 20](#_Toc145072845)

[13-Quaternion Anlatım ve İşlevleri 20](#_Toc145072846)

[14-Object Instantiate-Obje Oluşturma 20](#_Toc145072847)

[Bölüm-8: Unity Mathf-Hide Flags-Çeşitli İpuçları-Değişken Metodları 20](#_Toc145072848)

[1-Mathf-Matematiksel İşlemler 20](#_Toc145072849)

[2-Değişkenler İçin Çeşitli Metodlar-Örnek ve Kullanımı 21](#_Toc145072850)

[3-Hide Flags Yöntemleri ve Çeşitli İpuçları-I 21](#_Toc145072851)

[4-Hide Flags Yöntemleri ve Çeşitli İpuçları-II 21](#_Toc145072852)

[Bölüm-9: Unity Physics-Fizik-2D-3D 21](#_Toc145072853)

[1-3D|Fizik İşlemleri-I 21](#_Toc145072854)

[2-3D|Fizik İşlemleri-II 21](#_Toc145072855)

[3-3D|Fizik İşlemleri-III 21](#_Toc145072856)

[4-3D|Fizik İşlemleri-IV 21](#_Toc145072857)

[5-3D|Fizik İşlemleri-V 21](#_Toc145072858)

[6-3D|Jointler-I 22](#_Toc145072859)

[7-3D|Jointler-II 22](#_Toc145072860)

[8-3D|Jointler-III 22](#_Toc145072861)

[9-2D|Fizik İşlemleri-I 22](#_Toc145072862)

[10-2D|Fizik İşlemleri-II 22](#_Toc145072863)

[11-2D|Jointler-I 22](#_Toc145072864)

[12-2D|Jointler-II 22](#_Toc145072865)

[13-2D|Jointler-III 22](#_Toc145072866)

[14-2D|Effectors-I 22](#_Toc145072867)

[15-2D|Effectors-II 22](#_Toc145072868)

[16-2D|Effectors-III 23](#_Toc145072869)

[17-Sorting and Layers 23](#_Toc145072870)

[Bölüm-10: Unity Graphic-Lightning-Işık-Camera-Material-Texture-Diğer İşlemler 23](#_Toc145072871)

[1-Material|Texture|Shader|Mesh Anlatım-I 23](#_Toc145072872)

[2-Material|Texture|Shader|Mesh Anlatım-II 23](#_Toc145072873)

[3-Material|Texture|Shader|Mesh Anlatım-III 23](#_Toc145072874)

[4-Material|Texture|Shader|Mesh Anlatım-IV 23](#_Toc145072875)

[5-Material|Texture|Shader|Mesh Anlatım-V 24](#_Toc145072876)

[6-Material|Texture|Shader|Mesh Anlatım-VI 24](#_Toc145072877)

[7-Material|Texture|Shader|Mesh Anlatım-VII 24](#_Toc145072878)

[8-Material|Texture|Shader|Mesh Anlatım-VIII 24](#_Toc145072879)

[9-Camera|Anlatım ve İşlemler-I 24](#_Toc145072880)

[10-Camera|Anlatım ve İşlemler-II 24](#_Toc145072881)

[11-Occlusion Culling|Frustum Culling Sistemi-I 24](#_Toc145072882)

[12-Occlusion Culling|Frustum Culling Sistemi-II 24](#_Toc145072883)

[13-Lighting-I 25](#_Toc145072884)

[14-Lighting-II 25](#_Toc145072885)

[15-Lighting-III 25](#_Toc145072886)

[16-Lighting-IV 25](#_Toc145072887)

[17-Skybox 25](#_Toc145072888)

[18-HD Render Pipeline|Universal Render Pipeline 25](#_Toc145072889)

[Bölüm-11: Unity Efekt 25](#_Toc145072890)

[1-Component Efektleri-Trial Renderer 25](#_Toc145072891)

[2-Line Renderer|Halo|Projector|Lens Flare 26](#_Toc145072892)

[3-Script Dosyası İle Efekt Kontrolü 26](#_Toc145072893)

[4-Particle System|Particle Effect-I 26](#_Toc145072894)

[5-Particle System|Particle Effect-II 26](#_Toc145072895)

[6-Particle System|Particle Effect-III 26](#_Toc145072896)

[7-Particle System|Particle Effect-IV 26](#_Toc145072897)

[8-Particle System|Particle Effect-V 26](#_Toc145072898)

[9-Visual Graphic Effect-I 26](#_Toc145072899)

[10-Visual Graphic Effect-II 26](#_Toc145072900)

[11-Visual Graphic Effect-III-IV 26](#_Toc145072901)

[Bölüm-12: Unity UI-Canvas-Layout İşlemleri 26](#_Toc145072902)

[1-Canvas UI Genel Anlatım 26](#_Toc145072903)

[2-Canvas Elemanları-I 26](#_Toc145072904)

[3-Canvas Elemanları-II 26](#_Toc145072905)

[4-Canvas Elemanları-III 27](#_Toc145072906)

[5-Canvas Elemanları-IV 27](#_Toc145072907)

[6-Canvas Elemanları-V 27](#_Toc145072908)

[7-Canvas Elemanları-VI 27](#_Toc145072909)

[8-Canvas Elemanları-TextMeshPro Anlatım ve Ayarları 27](#_Toc145072910)

[9-Layout-Öğe Düzenleme-Layout Çeşitleri ve Ayarları 27](#_Toc145072911)

[Bölüm-13: Unity Animasyon 27](#_Toc145072912)

[1-Animasyon İşlemleri Anlatım ve Ayarlar-I 27](#_Toc145072913)

[2-Animasyon İşlemleri Anlatım ve Ayarlar-II 27](#_Toc145072914)

[3-Animasyon İşlemleri Anlatım ve Ayarlar-III 27](#_Toc145072915)

[4-Animasyon İşlemleri Anlatım ve Ayarlar-IV 27](#_Toc145072916)

[5-Animasyon İşlemleri Anlatım ve Ayarlar-V 28](#_Toc145072917)

[6-Animasyon İşlemleri Anlatım ve Ayarlar-VI 28](#_Toc145072918)

[7-Animasyon İşlemleri Anlatım ve Ayarlar-VII 28](#_Toc145072919)

[8-Örnek-3D Animasyon Paket-I 28](#_Toc145072920)

[9-Örnek-3D Animasyon Paket-II 28](#_Toc145072921)

[10-Örnek-2D Animasyon …-Sprite İle Animasyon Yapma 28](#_Toc145072922)

[Bölüm-14: Unity Audio-Ses 28](#_Toc145072923)

[1-Ses Componentleri ve Ses Ayarları-I 28](#_Toc145072924)

[2-Ses Componentleri ve Ses Ayarları-II 28](#_Toc145072925)

[3-Ses Componentleri ve Ses Ayarları-III 29](#_Toc145072926)

[Bölüm-15: Unity AI-Yapay Zeka-Navigation 29](#_Toc145072927)

[1-Tüm İşlemler-Anlatım ve Ayarlar-I 29](#_Toc145072928)

[2-Tüm İşlemler-Anlatım ve Ayarlar-II 29](#_Toc145072929)

[3-Tüm İşlemler-Anlatım ve Ayarlar-III 29](#_Toc145072930)

[4-Tüm İşlemler-Anlatım ve Ayarlar-IV 29](#_Toc145072931)

[5-Tüm İşlemler-Anlatım ve Ayarlar-V 29](#_Toc145072932)

[6-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-I 29](#_Toc145072933)

[7-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-II 30](#_Toc145072934)

[8-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-III 30](#_Toc145072935)

[9-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-IV 30](#_Toc145072936)

[10-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-V 30](#_Toc145072937)

[11-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-VI 30](#_Toc145072938)

[Bölüm-16: Unity Input-Kontrol Yöntemleri-Klavye-Mouse-Joystick-Touch Dokunma 30](#_Toc145072939)

[1-Input Manager-İşlemler-I 30](#_Toc145072940)

[2-Input Manager-İşlemler-II 30](#_Toc145072941)

[3-Input Manager-İşlemler-III 30](#_Toc145072942)

[4-Input Manager-Touch İşlemleri-I 30](#_Toc145072943)

[5-Input Manager-Touch İşlemleri-II 30](#_Toc145072944)

[6-Input Manager-Acceleration-Telefon Hareketlerini Yakalama 30](#_Toc145072945)

[Bölüm-17: Unity Diğer Çeşitli İşlemler-Çeşitli Yöntemler-İpuçları 31](#_Toc145072946)

[1-3D-Terrain Anlatım ve İşlemler-I 31](#_Toc145072947)

[2-3D-Terrain Anlatım ve İşlemler-II 31](#_Toc145072948)

[3-3D-Terrain Anlatım ve İşlemler-III 31](#_Toc145072949)

[4-3D-Terrain Anlatım ve İşlemler-IV 31](#_Toc145072950)

[5-3D-Terrain Anlatım ve İşlemler-V 31](#_Toc145072951)

[6-3D-Terrain Anlatım ve İşlemler-VI 31](#_Toc145072952)

[7-3D-Terrain Anlatım ve İşlemler-VII 31](#_Toc145072953)

[8-Past Processing-I 31](#_Toc145072954)

[9-Past Processing-II 31](#_Toc145072955)

[10-3D-2D RayCast Sistemi Anlatım ve İşlemler-I 31](#_Toc145072956)

[11-3D-2D RayCast Sistemi Anlatım ve İşlemler-II 32](#_Toc145072957)

[12-3D-2D RayCast Sistemi Anlatım ve İşlemler-III 32](#_Toc145072958)

[13-PlayerPrefs İşlemleri-Load&Save Sisteminin Tüm Detayları ve Örnekleri-I 32](#_Toc145072959)

[14-PlayerPrefs İşlemleri-Load&Save Sisteminin Tüm Detayları ve Örnekleri-II 32](#_Toc145072960)

[15-Dosya Sistemi İle Save&Load İşlemi-I 32](#_Toc145072961)

[16-Dosya Sistemi İle Save&Load İşlemi-II 32](#_Toc145072962)

[17-2D-Tile Map-Tile Palette-Anlatım ve İşlemleri-I 32](#_Toc145072963)

[18-2D-Tile Map-Tile Palette-Anlatım ve İşlemleri-II 32](#_Toc145072964)

[19-Genel İşlem-Unity Event System-Event Trigger-Drag-Click vb. İşlemler 32](#_Toc145072965)

[20-Miscellaneous İşlemler-Çeşitli İşlemler-Video Player 32](#_Toc145072966)

[21-Sahne Yönetim İşlemleri-SceneManager Komutları ve Örnekler 33](#_Toc145072967)

[22-Unity Standart Asset Paket İncelemesi 33](#_Toc145072968)

[Bölüm-18: Unity Çeşitli Örnekler-İpuçları 33](#_Toc145072969)

[1-FPS Anlatım ve Örnekler-I 33](#_Toc145072970)

[2- FPS Anlatım ve Örnekler-II 33](#_Toc145072971)

[3-Sahne Geçiş-Loading Ekranı Yapımı-Slider Ayarları 33](#_Toc145072972)

[4-ScrollView Örneği-Oyunlardaki Marketlerde Kullanılan Sistem 33](#_Toc145072973)

[5-Object Pooling-Obje Havuzu Yöntemi 33](#_Toc145072974)

[6-Ragdoll Sistemi Nedir-Ragdoll Oluşturulması 33](#_Toc145072975)

[7-Oyunlardaki Mini Map-Anlatım ve Yapımı 33](#_Toc145072976)

[8-Way Point-Anlatım ve Örneği 34](#_Toc145072977)

[9-Oyunlardaki El Feneri sistemi 34](#_Toc145072978)

[10-Oyunlardaki Tab Menü Sistemi 34](#_Toc145072979)

[11-Oyunlardaki Build(İnşa) Yapma-Obje Oluşturma Sistemi-I 34](#_Toc145072980)

[12-Oyunlardaki Build(İnşa) Yapma-Obje Oluşturma Sistemi-II 34](#_Toc145072981)

[13-Oyunlardaki Destruct(Obje Yıkma-Parçalama) İşleminin Yapılması-I 34](#_Toc145072982)

[14-Oyunlardaki Destruct(Obje Yıkma-Parçalama) İşleminin Yapılması-II 34](#_Toc145072983)

[15-Oyunlardaki Kumaş&Bayrak Efekti-Cloth Component-I 34](#_Toc145072984)

[16- Oyunlardaki Kumaş&Bayrak Efekti-Cloth Component-II 34](#_Toc145072985)

[17-Oyunlardaki Oyun İçi Grafik Ayarının Yapılabilmesi 34](#_Toc145072986)

[Bölüm-19: Multiplayer Sistemi-Online Oyun-Server İletişimi-Photon Sistemi 34](#_Toc145072987)

[1-Multiplayer Sistemi-Anlatım ve Örnekler-I 35](#_Toc145072988)

[2-Multiplayer Sistemi-Anlatım ve Örnekler-II 35](#_Toc145072989)

[3-Multiplayer Sistemi-Anlatım ve Örnekler-III 35](#_Toc145072990)

[4-Multiplayer Sistemi-Anlatım ve Örnekler-IV-V 35](#_Toc145072991)

[Bölüm-20: Unity Çeşitli Pencerelerin Yönetimi 35](#_Toc145072992)

[1-Project Settings-Audio-Editor-Unity Remote Sistemi-Remote 5 35](#_Toc145072993)

[2-Project Settings-Graphics-Physics 3D-Physics 2D 35](#_Toc145072994)

[3-Project Settings-Player-PC&MAC&LINUX Standalone-I 35](#_Toc145072995)

[4-Project Settings-Player-PC&MAC&LINUX Standalone-II 35](#_Toc145072996)

[5-Project Settings-Player-Android 35](#_Toc145072997)

[6- Project Settings-IOS-WebGL-Universal Windows 35](#_Toc145072998)

[7-Project Settings-Preset Manager-Quality 35](#_Toc145072999)

[8-Preferences Penceresi 35](#_Toc145073000)

[9-Unity Services Penceresi 35](#_Toc145073001)

[Bölüm-21: Unity Optimizasyon 36](#_Toc145073002)

[1-Script Optimizasyon İpuçları 36](#_Toc145073003)

[2-Profiller Penceresi-Genel Optimizasyon Konuları 36](#_Toc145073004)

[Bölüm-22: Unity Reklam-Oyun İçi Satın Alma Sistemi-Build İşlemleri 36](#_Toc145073005)

[1-2023 | OYUNLARA REKLAM EKLEME | ADMOB | PAKET KURULUMU | UYGULAMA OLUŞTURMA 36](#_Toc145073006)

[2-2023 | OYUNLARA REKLAM EKLEME | ADMOB | GEÇİŞ REKLAMI 36](#_Toc145073007)

[3-2023 | OYUNLARA REKLAM EKLEME | ADMOB | ÖDÜLLÜ REKLAM | ÖDÜLLÜ GECİŞ REKLAMI 37](#_Toc145073008)

[4-2023 | OYUNLARA REKLAM EKLEME | ADMOB | BANNER REKLAMI 37](#_Toc145073009)

[5-OYUN İÇİN SATIN ALMA EKLEME İŞLEMİ | GERÇEK UYGULAMA YÜKLEME VE TEST İŞLEMLERİ 1 37](#_Toc145073010)

[6-OYUN İÇİN SATIN ALMA EKLEME İŞLEMİ | GERÇEK UYGULAMA YÜKLEME VE TEST İŞLEMLERİ 2 37](#_Toc145073011)

[7-BUİLD ALMA | OYUNU PAKETLEME | GOOGLE PLAY | ANROİD 37](#_Toc145073012)

[8-GOOGLE PLAY'E UYGULAMA YÜKLEME ANLATIM | ÜRETİM | BETA | ALPHA KANALLARI 37](#_Toc145073013)

[9-BUİLD ALMA | OYUNU PAKETLEME | PC | MAC | WEBGL 37](#_Toc145073014)

[10-BUİLD ALMA | OYUNU PAKETLEME | İOS 37](#_Toc145073015)

[Bölüm-23: Ücretsiz Assets Bulabileceğin Web Siteleri 37](#_Toc145073016)

[1-Genel Anlatım ve Açıklama-Genel Asset-Model-Ses-Texture 37](#_Toc145073017)

[Bölüm-24: Örnek Oyun-3D-PC-FPS-Mahalleni Koru 38](#_Toc145073018)

[1-Mahalleni Koru 1 - Proje Oluşturma | Materyalleri ekleme 38](#_Toc145073019)

[2-Mahalleni Koru 2 - Sahnenin ve Mahallemizin oluşturulması 38](#_Toc145073020)

[3-Mahalleni Koru 3 - Karakterimizi ekleme | Crosshair | Ateş etme | Ses | Efekt 38](#_Toc145073021)

[4-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 1.KAYNAK DOSYASI 39](#_Toc145073022)

[5-Mahalleni Koru 4 – Mermi izi | Kan efekti | Vurulan objeye fizik uygulama 39](#_Toc145073023)

[6-Mahalleni Koru 5 - Silah Paketi yükleme | Animasyon Yapma 39](#_Toc145073024)

[7-Mahalleni Koru 6 - Animasyon Yapımı Devam 39](#_Toc145073025)

[8-Mahalleni Koru 7 - Şarjör ve Mermi Sistemi yapımı 39](#_Toc145073026)

[9-Mahalleni Koru 8 - Şarjör ve Mermi Sistemi yapımı 2 39](#_Toc145073027)

[10-Mahalleni Koru 9 - Boş Mermi Kovanı Çıkarma | Yeni Kaynak ekleme 39](#_Toc145073028)

[11-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 1 | Mermi Kutusu 1 40](#_Toc145073029)

[12-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 1 | Mermi Kutusu 2 40](#_Toc145073030)

[13-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 1 | Mermi Kutusu 3 40](#_Toc145073031)

[14-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 1 | Mermi Kutusu 4 40](#_Toc145073032)

[15-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 2 | Mermi Kutusu 5 40](#_Toc145073033)

[16-Mahalleni Koru 11 - PlayerPref sistemini kurma | Mermi kaydetme 40](#_Toc145073034)

[17-Mahalleni Koru 12 - (İsteğe Bağlı Part 1)Diğer silahların eklenmesi ve Animasyon 41](#_Toc145073035)

[18-Mahalleni Koru 13 - (İsteğe Bağlı Part 2)Diğer silahların eklenmesi ve Animasyon 41](#_Toc145073036)

[19-Mahalleni Koru 14 - (İsteğe Bağlı Part 3)Diğer silahların eklenmesi ve Animasyon 41](#_Toc145073037)

[20-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 2.KAYNAK DOSYASI 41](#_Toc145073038)

[21-Mahalleni Koru 15 - Sniper Silahı Scope (Yakınlaştırma) işlemini yapma 41](#_Toc145073039)

[22-Mahalleni Koru 16-AK47 | Magnum | Pompali | Scope (Yakınlaştırma) işlemini yapma 41](#_Toc145073040)

[23-Mahalleni Koru 17 Kod Revize işlemleri | Camera Shake (Kamera Titremesi) yapımı 41](#_Toc145073041)

[24-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 3.KAYNAK DOSYASI 41](#_Toc145073042)

[25-Mahalleni Koru 18 - Silah değiştirebilme sisteminin yapılması 41](#_Toc145073043)

[26-Mahalleni Koru 19 - Yapay Zekanın oluşturulması | Düşmanların İşlemleri 1 42](#_Toc145073044)

[27-Mahalleni Koru 20 - Yapay Zekanın oluşturulması | Düşmanların İşlemleri 2 42](#_Toc145073045)

[28-Mahalleni Koru 21 - Yapay Zekanın oluşturulması | Düşmanların İşlemleri 3 42](#_Toc145073046)

[29-Mahalleni Koru 22 - Yerden Sağlık alabilme işleminin yapılması 43](#_Toc145073047)

[30-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 4.KAYNAK DOSYASI 43](#_Toc145073048)

[31-Mahalleni Koru 23 - Toplam Düşman Sayısı işlemini yapma | SKYBOX EKLEME | DİĞER 43](#_Toc145073049)

[32-Mahalleni Koru 24 - Düşman karakterlerini ekleme 43](#_Toc145073050)

[33-Mahalleni Koru 25 - Düşman Animasyonlarının Yapılması | Koşma ve Ölme 43](#_Toc145073051)

[34-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 5.KAYNAK DOSYASI 43](#_Toc145073052)

[35-Mahalleni Koru 26 - Genel İyileştirmeler | Zombilere ve Oyun içine Ses ekleme 43](#_Toc145073053)

[36-Mahalleni Koru 27 - Bomba Atabilme özelliğinin Entegresi 1 43](#_Toc145073054)

[37-Mahalleni Koru 28 - Bomba Atabilme özelliğinin Entegresi 2 44](#_Toc145073055)

[38-Mahalleni Koru 29 - Item (Sağlık ve Bomba) Toplama ve Kullanma sistemi 1 44](#_Toc145073056)

[39-Mahalleni Koru 30 - Item (Sağlık ve Bomba) Toplama ve Kullanma sistemi 2 44](#_Toc145073057)

[40-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 6.KAYNAK DOSYASI 44](#_Toc145073058)

[41-Mahalleni Koru 31 - Çeşitli Ayarlar & Düzenlemeler 44](#_Toc145073059)

[42-Mahalleni Koru 32 - Çeşitli Ayarlar & Düzenlemeler 44](#_Toc145073060)

[43-Mahalleni Koru 33 - Mermi Çıkma ve gitme işleminin yapılması 44](#_Toc145073061)

[44-Mahalleni Koru 34 - Bazı işlemlerin geliştirilmesi 44](#_Toc145073062)

[45-Mahalleni Koru 35 - Oyun sonu panellerinin değiştirilmesi 45](#_Toc145073063)

[46-Mahalleni Koru 36 - Çeşitli iyileştirmelerin ve düzeltmelerin yapılması 45](#_Toc145073064)

[47-Mahalleni Koru 37 - Ana Menü Yapımı 45](#_Toc145073065)

[48-Mahalleni Koru 38 - Loading ve Oyundan çıkış panel yapımı 45](#_Toc145073066)

[49-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 7.KAYNAK DOSYASI + BUİLD ALINMIŞ HALİ 45](#_Toc145073067)

[Bölüm-25: Örnek Oyun-2D-Mobil-Multi Player Online Oyun-Kutuma Dokunma 45](#_Toc145073068)

[1-Kutuma Dokunma-I 45](#_Toc145073069)

[2-Kutuma Dokunma-II 45](#_Toc145073070)

[3-Kutuma Dokunma-III 45](#_Toc145073071)

[4-Kutuma Dokunma-IV 46](#_Toc145073072)

[5-Kutuma Dokunma-V 46](#_Toc145073073)

[6-Kutuma Dokunma-VI 46](#_Toc145073074)

[7-Kutuma Dokunma-VII 46](#_Toc145073075)

[8-Kutuma Dokunma-VIII 46](#_Toc145073076)

[9-Kaynak Dosya-I 46](#_Toc145073077)

[10-Kutuma Dokunma-IX 46](#_Toc145073078)

[11-Kutuma Dokunma-X 46](#_Toc145073079)

[12-Kutuma Dokunma-XI 46](#_Toc145073080)

[13-Kutuma Dokunma-XII 46](#_Toc145073081)

[14-Kutuma Dokunma-XIII 46](#_Toc145073082)

[15-Kutuma Dokunma-XIV 47](#_Toc145073083)

[16-Kutuma Dokunma-XV 47](#_Toc145073084)

[17-Kutuma Dokunma-XVI 47](#_Toc145073085)

[18-Kutuma Dokunma-XVII 47](#_Toc145073086)

[19-Kutuma Dokunma-XVIII 47](#_Toc145073087)

[20-Kaynak Dosya-II 47](#_Toc145073088)

[21-Kutuma Dokunma-XIX 47](#_Toc145073089)

[22-Kutuma Dokunma-XX 47](#_Toc145073090)

[23-Kutuma Dokunma-XXI 47](#_Toc145073091)

[24-Kutuma Dokunma-XXII 47](#_Toc145073092)

[25-Kutuma Dokunma-XXIII 48](#_Toc145073093)

[26-Kutuma Dokunma-XXIV 48](#_Toc145073094)

[27-Kaynak Dosya-III 48](#_Toc145073095)

[28-Kutuma Dokunma-XXV 48](#_Toc145073096)

[29-Kutuma Dokunma-XXVI 48](#_Toc145073097)

[30-Kutuma Dokunma-XXVII 48](#_Toc145073098)

[31-Kutuma Dokunma-XXVIII 48](#_Toc145073099)

[32-Kaynak Dosya-IV 48](#_Toc145073100)

[Bölüm-26: Örnek Oyun-3D-PC-Third Person-Suikast 49](#_Toc145073101)

[1-Suikast 1 49](#_Toc145073102)

[2-Suikast 2 49](#_Toc145073103)

[3-Suikast 3 49](#_Toc145073104)

[4-Suikast 4 49](#_Toc145073105)

[5-Suikast 5 49](#_Toc145073106)

[6-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 1.FULL KAYNAK DOSYASI 49](#_Toc145073107)

[7-Suikast 6 49](#_Toc145073108)

[8-Suikast 7 49](#_Toc145073109)

[9-Suikast 8 49](#_Toc145073110)

[10-Suikast 9 49](#_Toc145073111)

[11-Suikast 10 49](#_Toc145073112)

[12-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 2.FULL KAYNAK DOSYASI 49](#_Toc145073113)

[13-Suikast 11 49](#_Toc145073114)

[14-Suikast 12 49](#_Toc145073115)

[15-Suikast 13 49](#_Toc145073116)

[16-Suikast 14 49](#_Toc145073117)

[17-Suikast 15 49](#_Toc145073118)

[18-Suikast 16 49](#_Toc145073119)

[19-Suikast 17 49](#_Toc145073120)

[20-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 3.FULL KAYNAK DOSYASI 49](#_Toc145073121)

[21-Suikast 18 49](#_Toc145073122)

[22-Suikast 19 49](#_Toc145073123)

[23-Suikast 20 49](#_Toc145073124)

[24-Suikast 21 49](#_Toc145073125)

[25-Suikast 22 49](#_Toc145073126)

[26-Suikast 23 49](#_Toc145073127)

[27-Suikast 24 49](#_Toc145073128)

[28-Suikast 25 49](#_Toc145073129)

[29-Suikast 26 49](#_Toc145073130)

[30-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 4.FULL KAYNAK DOSYASI 49](#_Toc145073131)

[31-Suikast 27 49](#_Toc145073132)

[32-Suikast 28 49](#_Toc145073133)

[33-Suikast 29 49](#_Toc145073134)

[34-Suikast 30 49](#_Toc145073135)

[35-Suikast 31 49](#_Toc145073136)

[36-Suikast 32 50](#_Toc145073137)

[37-Suikast 33 50](#_Toc145073138)

[38-Suikast 34 50](#_Toc145073139)

[39-Suikast 35 50](#_Toc145073140)

[40-Suikast 36 50](#_Toc145073141)

[41-Suikast 37 50](#_Toc145073142)

[42-Suikast 38 50](#_Toc145073143)

[43-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 5.FULL KAYNAK DOSYASI 50](#_Toc145073144)

[Bölüm-27: Örnek Oyun-2D-Mobil-Hafıza Oyunu-Oyun Adı:Ben Bulurum 50](#_Toc145073145)

[1-Ben Bulurum-I 50](#_Toc145073146)

[2-Ben Bulurum-II 50](#_Toc145073147)

[3-Ben Bulurum-III 50](#_Toc145073148)

[4-Ben Bulurum-IV 50](#_Toc145073149)

[5-Kaynak Dosya-I 50](#_Toc145073150)

[6-Ben Bulurum-V 50](#_Toc145073151)

[7-Ben Bulurum-VI 50](#_Toc145073152)

[8-Ben Bulurum-VII 50](#_Toc145073153)

[9-Ben Bulurum-VIII 51](#_Toc145073154)

[10-Ben Bulurum-IX 51](#_Toc145073155)

[11-Kaynak Dosya-II 51](#_Toc145073156)

[12-Ben Bulurum-X 51](#_Toc145073157)

[13-Ben Bulurum-XI 51](#_Toc145073158)

[14-Kaynak Dosya-III 51](#_Toc145073159)

[Bölüm-28: Örnek Oyun-3D-PC-Araba Yarışı-Speedy Gonzales 51](#_Toc145073160)

[1- 52](#_Toc145073161)

[2- 52](#_Toc145073162)

[3- 52](#_Toc145073163)

[4- 52](#_Toc145073164)

[5- 52](#_Toc145073165)

[6- 52](#_Toc145073166)

[7- 52](#_Toc145073167)

[8- 52](#_Toc145073168)

[9- 52](#_Toc145073169)

[10- 52](#_Toc145073170)

[11- 52](#_Toc145073171)

[12- 52](#_Toc145073172)

[13- 52](#_Toc145073173)

[14- 52](#_Toc145073174)

[15- 52](#_Toc145073175)

[Bölüm-29: Bonus 52](#_Toc145073176)

[1- 52](#_Toc145073177)

[2- 52](#_Toc145073178)

[3- 52](#_Toc145073179)

[4- 52](#_Toc145073180)

[5- 52](#_Toc145073181)

[6- 52](#_Toc145073182)

[7- 52](#_Toc145073183)

[8- 52](#_Toc145073184)

[9- 52](#_Toc145073185)

[10- 52](#_Toc145073186)

[11- 52](#_Toc145073187)

[12- 52](#_Toc145073188)

[13- 52](#_Toc145073189)

[14- 52](#_Toc145073190)

[15- 52](#_Toc145073191)

[Alfa-Beta-Gama 53](#_Toc145073192)

[1-Alfa 53](#_Toc145073193)

[2-Beta 53](#_Toc145073194)

[3-Gama 54](#_Toc145073195)

# Bölüm-1: Genel Bilgilendirme

## 1-Tanıtım

\*Kurs sıfırdan.

## 2-Örnek Oyun Kazanç Tablosu

* Oyun fikrine başlamadan önce rakip analizi çok önemlidir.
* Bilardo oyunu örneğini verdi. Büyük firmalar, bu örnek gibi zaten tutmuş oyunlarına milyonlarca lira reklam ayırıyor. Para kazanmak istiyorsan ya rekabetin az olduğu alanlara yöneleceksin veya çok orijinal fikirlerin olacak.
* Oyunlarını reklama asla boğma. Oyuncular oyunu sevseler bile oyunu anında silerler. Oyuncu dostu reklamlar koymalısın.

## 3-3D-2D Oyun Nedir? Aralarındaki Fark Nedir?

* Nesnelerin tüm boyutlarının göründüğü oyun türü 3D’dir.
* 2D oyun türlerinde derinlik (Z ekseni) bulunmaz. Yalnız X ve Y ekseni bulunur.
* 2D oyunlar genellikle mobildir.
* 2D oyunlarda animasyon için ya yukarı aşağı veya sağa sola hareket vardır. Ancak 3D oyunlarda buna bir eksen ve dönme de eklenir. 3D’ de animasyon çok daha zor ve detaylıdır.
* \*A :: 2.5 D, 2 boyut kamera açısına sahip, 3 boyutlu tasarlanmış oyun.

## 4-Oyun Türleri Nelerdir?

* TPS: Third Person Shooter
* MOBA: Multiplayer Online Battle Arena (Takım savaşı)
* RTS: Real Time Strategy
* TBS: Sıra tabanlı strateji (Satranç)
* Platform: Bir platformda ilerleme, genellikle mobil.

## 5-Merak Edilen Sorular ve Cevapları

* \*G :: Yeni öğrenenlerin yaptıkları en büyük hata, acele etmektir. Oyun geliştirmeyi kolay olarak düşünüp hızlı para kazanmak istersen kaybedersin. Asla acele etme.

# Bölüm-2: Unity Kurulum ve Tanıtım

## 1-Unity Hub Kurulum

* Unity HUB, Unity ile bizim aramızdaki bir arayüzdür. Tutorial içerir.
* Birçok sürüm seçeneği var çünkü proje başladığı sürümden farklı bir sürümle çalışırsa hatalar oluşabilir.

## 2-Unity Arayüz Tanıtımı ve Asset Store

Paket yönetimi için Package Manager kullan. Çünkü eski yöntem tam silmeyerek sorunlara sebep oluyordu. (Sorun-Silme-Package Manager)

# Bölüm-3: Unity-Temel C# Dersleri

## 1-Değişken Tanımlama Türleri

* Preference’dan cs scripti ne ile açmak istediğini ayarlayabilirsin.
* Float tanımlamak için :: float küsurat=3.55f;
* Bool değerler tanımlanır ve atanmazsa false olur.

## 2-If-Else

## 3-For Döngüsü

## 4-Array

* Diziyi elemanlarıyla tanımlama :: int[] sayilarim=new int[]{1,2,4,8};
* Daha kısası :: int[] sayilarim={1,5,19};

## 5-Foreach

## 6-While-Do Döngüsü

## 7-Switch-Case

* Default burada else gibi çalışır.
* Burada case ”Audi”:

case “BMW”

dersen bu || gibi davranır. (Switch-Case’de ||)

## 8-List Kavramı

* \*G :: ArrayList’in kullanılması artık önerilmiyor. List<> kullan.

## 9-Tuple Patterns(İleri Seviye)

* \*B :: #region kullan. Kodun çok daha derli toplu gözükür.

## 10-Property Patterns(İleri Seviye)

* Lambda’da else :: \_=>0

## 11-Properties Kullanımı(İleri Seviye)

* Altin adlı consun olsun. Altin=20 dersen setter çalışır.

# Bölüm-4: Unity Arayüz Tanıtımları ve Çeşitli İşlemler

## 1-Sürüm ve Platform Yönetimi

* Build Settings’ten platform değiştirebilirsin. Yani Android için de uygulama geliştirebilirsin.

## 2-Console Penceresi Anlatımı-Debug Çeşitleri

* Collapse, aynı türde olan mesajları veya uyarıları tek çatı altında toplar.
* LogWarning ve LogError ile kendi uyarı ve hatalarını verdirebilirsin.

## 3-Scene ve Game Pencereleri

* DrawMode, Gizmos ve pusula kullanımı. Search kullanımı (Scene için).
* Game’deki Portrait dikey, Landscope yatay versiyondur. Bunu kullanmazsan kayma ve görüntü bozulması yaşayabilirsin.

# Bölüm-5: Unity Fonksiyonları ve Kodları

## 1-Class-Fonksiyon-I-II-III-IV

* Start fonksiyonu MonoBehaviour’dan miras.
* Eğer bir metodu protected yaparsan, onu yalnızca child classtan direkt adıyla çağırabilirsin.

## 6-Using Kullanımı-Namespace

* Namespace, büyük projelerde class adlarının çakışmasını engeller.
* Namepace ayrıca classları toplar.
* Namespace’i using ile dahil ettiğinde, içindeki classları kullanabilirsin.

## 7-MonoBehaviour-Messages

(Temel Fonksiyonlar)

* Shift+sağ tık :: Oraya kadarkilerin hepsini seçer.
* Awake, ilk çalışan fonksiyondur. OnEnable nesne aktif ise, OnDisable nesne aktif değilse çalışacaktır.
* Awake, OnEnable, Start oyun çalıştığında 1 kez çalışan, tekrar etmeyen fonksiyonlardır.
* Scriptteki fonksiyonların hepsini kullanmak zorunda değilsin. Ancak kullanacaksan sırası önemli. Örnek scriptteki gibi sıralanmalılar.
* Update, her karede, yani oyun aktif olduğu müddetçe çalışır. Bu fonksiyonu çok fazla kullanmak performansı düşürür. Yani Update’te yazılmadan çözülebilecekse o şekilde çözmelisin.
* FixedUpdate’te ise çalışma hızını ayarlayabilirsin Update’e ek olarak. Bunu Edit-Project Settings-Time da görebilirsin. Yanlış kullanımı Update’in yanlış kullanımından da kötü sonuç verir. (Fixed Update)
* LateUpdate, sahnedeki objelerin bütün Update metotlarının çalışması bittikten sonra çalışır.
* \*B :: Objeyi pasif yapmak istiyorsan üstteki checkbox’taki tiki kaldır. Oyun çalışırken pasif yaparsan OnDisable çalışır.

## 8-MonoBehaviour-Coroutine

* Coroutine Update’e benzer ancak Coroutine belirlediğimiz zaman aralıklarında çalışır ve istenildiği zaman durdurulup başlatılabilir.
* Bazı sistemlerde Coroutine kullanılır yalnızca, yani Update kullanılmaz. Hatta böyle yapmanın performansı iyileştirdiğini söyleyenler vardır. Yapısı ::

IEnumerator Deneme()

{

//kodlar

//bekleme zamanı

//istenirse diğer kodlar

}

* yield return null :: Bir kare bekle demektir. Update gibi olur.
* Coroutine’yi oluşturman yetmez, çağırmalısın. Bunu StartCoroutine(CoroutineAdi) ile gerçekleştirebilirsin.
* yield return new WaitForSeconds(3) :: 3 saniye bekler.
* \*B :: Coroutine’u dizayn ettikten sonra private değişkene atayarak kullan. (Coroutine’yi Privateye Ata)
* Update ile, içindeki kod sayesinde ‘space’ yapıldığında Coroutine durur.
* StopAllCoroutines() ile çalışan bütün coroutine’leri durdurabilirsin.
* WaitForFixdUpdate() :: Bütün scriptlerdeki FixedUpdate’lerin bitmesini bekler.

## 9-MonoBehaviour-Invoke

* Invoke, bizim verdiğimiz bir fonksiyonu belli bir süre çalıştırır.
* CancelInvoke() ile tümünü durdurabilirsin.
* IsInvoke(“InvokeAd”) ile bir invoke’un çalışıp çalışmadığını kontrol edebilirsin.

# Bölüm-6: Unity’ye İlk Adımlar-Çeşitli Obje İşlemleri-Isınma Turları

## 1- Objelerin Anlatımı-Scale-Move-Rotate ve Diğer Detaylar

* UI = Canvas :: Kullanıcının etkileşebildiği her şey.

## 2-Inspector Detayları ve Component Anlatımları-I

* Component :: Bir objenin ses, ışık, fizik olayı vb. özellikleri.
* Materyali 2 yolla objeye atayabilirsin :: Üstüne veya Mesh Renderer’e atarak.
* \*A :: Nesneyi pasif yaparsan ona atanmış script çalışmaz.

## 3-Inspector Detayları ve Component Anlatımları-II

* Statik objenin dezavantajı, oyunun boyutunu arttırmasıdır.
* Layer genelde 2D oyunlarda kullanılır. Hangi nesnenin önde-arkada olacağını belirler.

## 4-Inspector Detayları ve Component Anlatımları-III

* Preset oluşturma ve component detaylarını gördük. Topluca (birçok obje seçiliyken) ayarlarını değiştirmeyi gördük.

## 5-Boş Obje Oluşturma ve Görevleri-Hierarchy Child

* Add Component’e direkt scriptinin adını yazarak script oluşturabilirsin.
* Bir objeye birçok script atayabilirsin.
* Script dosyası, objelere görev verebilmeni sağlar.
* Arka planda yapılacak işlemleri boş objeye atarsın.
* Aynı zamanda boş objeye diğer objeler sürükle-bırak yaparak obje grupları oluşturabilirsin.

## 6-Objeye Değişken Tanımlama ve Çeşitli İşlemler

* Public proplar oluşturduk ve bunlara Inspector’dan değer ataması yaptık.

## 7-Array-Diziler ve List Anlatım ve Tanımlama

* Listeler dizilerden çok daha esnektir.

## 8-GetComponent Anlatımları ve İşlemleri-I

* \*B :: Component yakalamayı değişkene ata. Aksi halde performans kaybedersin.
* Bunu atadığın değişken ‘Capsule Collider’ gibi bir component türünde olmalı.
* Bir objede birden fazla aynı türde component (örneğin ses) olabilir. Bunları ‘GetComponents’ ile yakalayabilirsin. Bunları diziye atarak toplu işlem gerçekleştirebilirsin.
* \*G :: GetComponent, eğer aynı türde birden fazla component varsa ilkini yakalar.

## 9-GetComponent Anlatımları ve İşlemleri-II

* Child ve parent’lerin componentlerini yakalamak için ‘GetComponentInParent’ ve ‘GetComponentInChildren’ kullanılır.
* Bunda da ‘GetComponents…’ yapabilirsin.

## 10-Obje Yakalama-GameObject.Find-FindWithTag-FindGameObjectsWithTag-transform.Find-I

* Bir objeye atanmış scriptten başka bir objenin herhangi bir component’ini değiştirebilirsin.
* Yakaladığın Objenin alt ve üst soyuna da ‘…Find(“Ana/Cocuk”)…’ ile erişebilirsin.
* Find, performansı düşürür. Bu sebeple dikkatli kullanılmalı. Ayrıca Update içinde kullanılması asla tavsiye edilmez. Sürekli kullanacaksan GameObject türünde bir değişkene ata.
* \*B :: ‘FindWithTag’ı gördük. Daha iyi bir seçenektir. Aynı tagda birden çok taglı obje varsa bunlardan ilkini yakalar.

## 11-Obje Yakalama-GameObject.Find-FindWithTag-FindGameObjectsWithTag-transform.Find-II

* ‘transform.Find(…)’, odaktaki objenin 1.kuşak çocuklarında arama yapar. Performans için daha iyidir. (Soru-1)
* Unity’den, yani dışarıdan, obje aramak için ‘public GameObject’ yaratabilirsin.

## 12-GetComponent typeof-FindObjectOfType-SetActive-activeSelf

* gameObject, nesnenin kendisini temsiller. Bu şekilde GetComponent’i <> yerine () ile veya (typeof()) ile kullanabilirsin.
* \*B :: Bir obje pasifse hiçbir işlemini gerçekleştiremez. Buna kendini aktifleştirmek de dahildir.

## 13-CompareTag-AddComponent

* gameObject.CompareTag(“tagismi”) true-false döndürür.
* \*B :: <> = (typeof())

## 14-BroadCastMessage-SendMessage

* BroadCast’ın SendMessage’den farkı, eğer childlerde da aynı adda metot varsa onları da tetikler.

## 15-Gameobject Destroy-Component Destroy

* Destroy(GameObject,süre)
* Destroy(this) => Scripti yok eder.
* Destroy(GetComponent<Capsule Collider>()) => Componenti yok eder.

# Bölüm-7: Unity Transform&Vector&Quaternion İşlemler ve Kodlar-Obje Oluşturma

## 1-Proje-Import Package-Export Package

* Export ve Import ile tüm bölüme ayırma sorunlarını halledebilirsin.

## 2-Prefab Oluşturma ve Anlatımı

* Prefab, bir objeyi örneklendirme olarak saklamak ve ihtiyaç oldukça sahnelere eklemektir. Birinde yapılan değişiklik her sahnedeki objeye yansır. Elbette bunu daha sonra prefab’dan çıkararak sahneye özel hale getirebilirsin.

## 3-Transform Translate-Vector3 Static Özellikler-I

* Vektörlerde Vector4, genelde gölgelendirmede kullanılır.

## 4-Transform Translate-Vector3 Static Özellikler-II

* Vektörü kendin oluşturabileceğin gibi “Vector3…” ile hazır vektörler çağırabilirsin. Örneğin Vector3.One => Tüm değerleri 1’dir.
* \*B :: Rotasyon mevcutsa, düz ilerletmek istesen de, doğal olarak düz gitmez.

## 5-Vector3-Özellikler ve Public Metotlar

* transform.position.normalized :: Birim vektör verir. Elbette cisme ait.
* \*G :: ‘Equals’ metodu her zaman doğru çalışmaz vektörlerde. Vektörlerin ‘==’ ile karşılaştırılması gerekir. (Vektörler ve Equals)

## 6-Vector3-Static Methodlar-I

* Vector3.Angle(…)’a 2 vektörün farkını vermelisin.
* Vector3.Distance ‘yi gördük.

## 7-Vector3-Static Methodlar-II

* Time.time :: Oyun başından beri geçen süreyi verir.
* Time.deltaTime :: 1 karenin başlangıçtan bitişe geçen süresini verir.
* Time.timeScale= .5f :: Zamanı %50 yavaşlatır. Sıfıra eşitlersen oyun durur.
* transform.Translate(Vector,time) :: Objeyi hareket ettirir. (Soru-2)
* Vector3.Lerp’i gördük.

## 8-Vector3-Static Methodlar-III ve Operatörler

* İki vektörü Vector3.Scale ile çarpabilirsin, ancak \* ve / çalışmaz. Yani çalışır ama bir operand vektör, diğeri float olmalı.

## 9-Transform-Özellikler ve Methodlar-I

* \*A :: Space.Self’te hareket için kendini referans alır. Space.World’de çevreye göredir. Gerçek dünya gibi olanı world’dür.

## 10-Transform-Özellikler ve Methodlar-II

* Bir obje diğer bir objenin alt objesi ise, ana objenin pozisyonu değiştikçe pozisyonu değişecektir.
* ‘…transform.localXX’ te XX yerine Position veya Scale yazabilirsin ancak Rotation yazarsan değer vermez. Bu lokal pozisyonlar, ana objeye endekslidir.
* \*G :: transform.LookAt’ i gördük. Bu sanırım kameranın sürekli bakmasını sağlıyordu.

## 11-Transform-Özellikler ve Methodlar-III

* \*G ::transform.Find ile nesneyi yakaladın ve adını değiştirdin. Sonra değiştirdiğin çocuğuna ilk adı kullanarak ulaşamazsın. Değistirdiğin adla ulaşabilirsin.
* childCount yalnız çocukları sayar, torunları saymaz. hierarchyCount hepsini (kendisi dahil) sayar.
* GetChild(index) ile çocukları for ile dolaşabilirsin. Scriptte örneği mevcut.

## 12-Transform-Özellikler ve Methodlar-IV

* SetParent yaptığında child’ın pozisyonu sıfırlanır eğer 2. Parametreye false atamazsan.

## 13-Quaternion Anlatım ve İşlevleri

* Dönüş işlemi olduğunda, açılar arasında üst üste binmeler gibi sorunlar çıkar. Atlama ve sıçrama hataları oluşur. Quaternionlarda 4. Açı olduğundan daha stabil çalışır.
* Quaternion.Euler(…) istenen eksende döndürür. Update içine yazsan da işlemi 1 defa yapar.
* Objeyi kendi ekseninde döndüreceksen, FromToRotation kullan.
* Quaternion.identity ile objenin rotasyonunun sıfırlayabilirsin.
* ‘Quaternion.XXX’ yani değişik birkaç metot daha var.

## 14-Object Instantiate-Obje Oluşturma

* Instantiate(klanlanacak, parenti) parent vermek opsiyoneldir!
* \*G :: Fırlatılacak objeyi ‘prefab’ klasöründen atadım. Direkt var olan nesneyi oraya atarsan, nesne destroy edildikten sonra exception yersin.
* Mermi

# Bölüm-8: Unity Mathf-Hide Flags-Çeşitli İpuçları-Değişken Metodları

## 1-Mathf-Matematiksel İşlemler

* \*B :: Round metodu ‘.5f’ de sayı tekse veya çiftse farklı şekilde davranır.
* Mathf.Clamp(), değer – ise 0, + ise 1 döndürür.

## 2-Değişkenler İçin Çeşitli Metodlar-Örnek ve Kullanımı

* Trim(), baştaki ve sondaki boşlukları temizler. String metotudur. TrimStart ve TrimEnd versiyonları da vardır.

## 3-Hide Flags Yöntemleri ve Çeşitli İpuçları-I

* Hem public olmasını, hem de Inspector’da görünmemesini istiyorsan [HideInInspector] kullan.
* [Range(1,6)] ile girilebilecek değeri sınırlayabilirsin. Bununla aynı çalışan [Min()] de var. Ufak bir fark olarak Range’de slider mevcut.
* gameObject.hideFlags=HideFlags.XXX :: Temel amaç ‘Game’ ekranında işleri kolaylaştırmak.

## 4-Hide Flags Yöntemleri ve Çeşitli İpuçları-II

AddComponent’e dahi yeni şeyler ekleyebilirsin. Örneğin önceden oluşturduğun bir script dosyası. Bu, class’ın dışına yazılır. Burada mevcut menülere de referans verebilirsin, kendi menünü de oluşturabilirsin.

[RequireComponent()] ile, bir elemanı (scripti) eklediğinde, diğer componentlerin otomatik eklenmesini sağlayabilirsin. Hatta otomatik geleni silmeye çalıştığında hata alırsın script aktifse.

[Multiline(int)] :: String için yer açar.

[Tooltip(…)], [Header(…)], [ContextMenu(…)], [AddComponentMenu(…)] (Hide Flags)

# Bölüm-9: Unity Physics-Fizik-2D-3D

## 1-3D|Fizik İşlemleri-I

* Collider’deki materyal, diğer materyalden farklı. Bu fizik materyali.
* IsTrigger, çarpışan nesnelerden birinde açıksa, script tarafında bunu öğrenebilirsin.

## 2-3D|Fizik İşlemleri-II

* Fizik olaylarının gerçekleşmesi için RigidBody componentine ihtiyacın var.

## 3-3D|Fizik İşlemleri-III

* Rigidbody verdiğinde, nesnenin hareketini fizik motoruna bırakmış olursun.
* Çarpışmalarda titreşim görüyorsan Interpolate ile oyna. (Soru-3)

## 4-3D|Fizik İşlemleri-IV

* Oncollision ile çarpışma yakalamak istiyorsan, objelerden en az birinde Rigidbody olmalı.
* Ontrigger ise çarpışmayı Collider’den yakalar. Objelerden en az birinin IsTriggeri açık olmalı. IsTrigger verilerin fizik motoruna gönderilmesini sağlar.
* \*B :: .tag==”” yerine CompareTag() kullan.

## 5-3D|Fizik İşlemleri-V

* WheelCollider, Rigidbody olmadan çalışmaz.
* CharacterController’da SkinWidth sıkışmayı, Min Move Distance ise titremeyi engeller.

## 6-3D|Jointler-I

* Eğer Hinge Joint (Menteşe) kullanacaksan, kapının bağlı olduğu duvarın hareketini Rigidbody’den kısıtlamalısın.

## 7-3D|Jointler-II

* Zincir yaparken de Hinge Joint kullanabilirsin.

## 8-3D|Jointler-III

* SpringJoint ve Constant Force işlendi.

## 9-2D|Fizik İşlemleri-I

* RigidBody2D’de kinematic seçersen fizik motoru obje için çalışmaz.
* RigidBody2D’de Sleeping Mode ile güçten tasarruf edebilirsin. ‘Start Asleep’ ile nesneyi etkileşim olana kadar uykuda tutabilirsin.
* \*B :: Kinematic obje, kinematic veya statik objeler ile çarpışamaz. Sadece dinamik objelerle çarpışır. ‘Use Full Kinematic’ seçeneği ile bu durumu tüm çarpışmaları yakalayacak şekilde değiştirebilirsin.
* BoxCollider2D’deki Auto Tiling ile sprite’ın şekli değiştikçe collider buna uyum sağlar. (Soru-4)

## 10-2D|Fizik İşlemleri-II

* \*B :: Dinamic objeler tüm türlerdeki objelerle etkileşime girebilir.
* Edge Collider ile hazır colliderlerin sunmadığı şekilleri oluşturabilirsin. Şekil vermek için shift’e basılı tut.
* \*B :: Edge Collider içi boş, Polygon Collider içi dolu.
* \*B :: Composite Collider diğer colliderları birleştirmemizi sağlar. Herhangi bir colliderı dahil etmek için ‘Used by Composite’yi seç

## 11-2D|Jointler-I

* 2D Hinge Joint gördük. Diğer tür jointleri de inceledik. Bunlar Fixed, Friction ve Direction.

## 12-2D|Jointler-II

* Relative Joint 2D, bir nesneyi diğer bir nesneye bağlı sarkaç gibi yapar.
* Slider Joint ile bir noktadan bıraktığın cisim doğrusal hareket eder. Sürgülü kapı yapabilirsin bu jointle.

## 13-2D|Jointler-III

* Spring joint, Target joint, Wheel joint işlendi.

## 14-2D|Effectors-I

* Effector’da Force Target’te Collider seçersen cismi hem fırlatır hem de döndürür. Rigidbody olursa dönmez.
* Effector’da Drag’ı (Damping’de) arttırarak cismin önceden kazandığı hızı sönümleyebilirsin.

## 15-2D|Effectors-II

* Buoyancy (Su) Efectoru gördük.
* \*G :: IsTrigger, Box Collider’da açıkken, nesne diğer nesnelerle çarpışmaz. Ama çarpışmaları yakalar.

## 16-2D|Effectors-III

* Platform Efector’daki ‘Use One Way Grouping’ ile, Collider’i kompozit olan nesneler için daha etkin hesaplar yapabilirsin. Hepsini yakalar.
* Platfor Efector’de Side’ları kullanarak yandan çarpışmaları da ayarlayabilirsin.
* Platform Efector tam anlaşılmadı. Objenin havada kalmasını ve çarpışmalarını yakalamak istiyorsan kullanmalısın. (Soru-5)

## 17-Sorting and Layers

* Order in Layer’a eksi değer verebilirsin.
* Sorting Layer ise o etikete sahip objeler arasında sıralama yapar.

# Bölüm-10: Unity Graphic-Lightning-Işık-Camera-Material-Texture-Diğer İşlemler

## 1-Material|Texture|Shader|Mesh Anlatım-I

* Meshed (İskelet), Materials (Renk vb.), Shaders (Gölgelendirici), Textures (Doku) tanımları yapıldı.
* Materyal ekranında Shader türünü doğru seçmelisin. Örneğin mobile çıkacak bir oyunda standart seçersen telefonun ekran kartı yetmeyebilir.
* Shaderdeki ‘Standart Specular’, Standart’ın Metalic property’si içermeyen hali. Genelde yansımayla alakalı ayar yapılmayacaksa spec. daha uygundur.
* Nesneye koyacağın texture küçük olursa bozulmalar, büyük olursa yırtılmalar olur. O sebeple nesneye uygun bir boyut seçmelisin.
* \*A :: Texture’da metalik yoksa, metalik yerine ‘Roughness’i koyabilirsin.

## 2-Material|Texture|Shader|Mesh Anlatım-II

* Smoothness altındaki Source’de Albedo seçersen performanstan kazanırsın. Metalic ise boyutu büyük nesnelerin daha iyi görünmesini sağlar.

## 3-Material|Texture|Shader|Mesh Anlatım-III

* Occlusion, objede yarık-kırık varsa devreye girer ve o bölgedeki ışık kırılımını değiştirir.
* Specular Highlights’ı kapatırsan cisim için ışık hesaplanmaz, ancak performansı ciddi oranda arttırır. Mobil cihazlarda kullanılabilir.
* Emission’da ışık kaynağı yapabilirsin. ‘Global Illumination’daki seçeneklerle nelerin etkileneceğini seçebilirsin. ‘Realtime’ her şeye, ‘Baked’ yalnız statik nesnelere yayar. Performansı ciddi şekilde etkiler.

## 4-Material|Texture|Shader|Mesh Anlatım-IV

* Fade’in Transparent’ten farkı, yansımaların ve vurgulamaların yok olmasıdır. Işık almayacak ve karşı tarafı göstermesi istenilen objelerde kullanılır.
* Shader’daki Autodesk Interactive, Standart’tan da daha kalitelidir.

## 5-Material|Texture|Shader|Mesh Anlatım-V

* \*A :: Textures’in Inspector’unda aşağıdaki ‘Use Crunch Compression’, objeyi sıkıştırırken işlemciyi de devreye sokar.

## 6-Material|Texture|Shader|Mesh Anlatım-VI

* Texture’da Shape olarak Cube seçersen, yansımayla alakalı ek seçenekler gelir.
* \*B :: Read/Write Enabled’i işaretleyerek Texture özelliklerine scriptten erişebilirsin. İhtiyaç yoksa açmamalısın.
* \*B :: sRGB’yi çok iyi detaylanmış bir objen varsa kapat.
* Alpha Source genelde None seçilir. Böylece daha iyi performans verir.

## 7-Material|Texture|Shader|Mesh Anlatım-VII

* \*B :: Texture’deki ‘Aniso Level’, dokuya dik olarak yaklaştıkça kalitesinin artmasını sağlar. Arttırdıkça performans düşer.

## 8-Material|Texture|Shader|Mesh Anlatım-VIII

* Sprite Editor’de ‘Custom Outline’ kullanarak sprite’ın boşluğunu azaltabilirsin. Performans artışı sağlar.
* Sprite Editor ile ‘Custom Physics Outline’ ile fizik çerçevesi oluşturabilirsin.
* Sprite Mode’u multiple yaparak Sprite Editor’deki Slice’ı kullanabilirsin. Bununla sprite’ı parçalarsın ve bu parçalarla efekt yapabilirsin.
* Yine multiple’ı seçtiğinde ‘Secondary’ seçebilirsin. Bu sanırım animasyonla ilgili.

## 9-Camera|Anlatım ve İşlemler-I

* Projection’daki perspective, normal bakış açısıdır. Orthographic ise, perspektifin olmadığı, derinlik (yani uzaklığa bağlı değişim) içermeyen kamera açısıdır.
* Yine Camera’daki ‘Render Texture’ye ilgili RT nesnesini atayarak daha detaylı ayarlara erişebilirsin.
* MSAA, yaklaşıldığında objedeki pürüzleri azaltır.
* Dynamic Resolution ile FPS’de düşme olursa devreye girerek FPS’i kaliteyi düşürerek arttırır.

## 10-Camera|Anlatım ve İşlemler-II

* Kamerayı script ile kontrol etmeyi gördük. Fiziksel kamera da incelendi.

## 11-Occlusion Culling|Frustum Culling Sistemi-I

* Occlusion Culling, nesne arkasında kalan nesnelerin, GPU’ya gönderilmemesi. Frustum Culling ise mesafeye göre nesnelerin GPU’ya gönderilmemesi.
* \*B :: Duvar gibi objelerin arkasındaki objelerin, açıya girmeden GPU’ya gitmesi gereksizdir. Bu sebeple Inspector’daki obje adı yanındaki ‘Static’ kısmını ‘Occuluder Static’ seçmelisin.
* Bir üsttekinin çalışması için, gizlenecek objeyi ‘Occlude Static’ olarak ayarlamalısın.
* Eğer obje hareket etmeyecekse, Mesh’in altındaki ‘Dynamic Occ.’ı kapat.

## 12-Occlusion Culling|Frustum Culling Sistemi-II

* Occlusion/Bake’deki ‘Smallest Occluder’i iyi ayarlamalısın. Çok küçük yaparsan, küçük nesneleri de işler ve boyut artar. Çok büyük yaparsan performans düşer.
* Occlusion Portal ile kapı kapalıyken arkasındaki nesneleri GPU’ya göndermezsin. Open’ı kapatmalısın. Bu bir component’tir.
* Occlusion Area ile Occ. İşleminin uygulanacağı yerleri seçebilirsin. Performansı arttırır.

## 13-Lighting-I

* Directional Light’ın yeri değil açısı önemlidir.
* Area Light’ın çalışması için, Window’daki sekmesinde ‘Bake’ ederek hesaplamasını sağlamalısın. Etkilenmesini istediğin objeler statik olmalı. Ekranın sağ altında ‘Auto Gen. Lightning…’ var. Işıkla uğraşırken onu açabilirsin.

## 14-Lighting-II

* Area Light’ı kaldırsan bile hesaplamadan gitmez.
* Cookie’leri gördük.

## 15-Lighting-III

* Probe’lar, boşluklardaki ışıkları yönetmeni sağlar. Yakın ışıkları, kendine yakın objelere yansıtır.
* Lightning’in settinglerini inceledik. Window’dan buraya ulaşabileceğin gibi sağ alttaki ikondan da ulaşabilirsin.

## 16-Lighting-IV

* Lightmapping (Lightning Setupta) bake işlemiyle ilgilidir.

## 17-Skybox

* Lightning’e skybox koyup sonra skybox ile oynayabilirsin.
* Asset Store’dan indirebileceğin Skybox’ları,bir Skybox tarzı olan ‘CubeMap’e koyarak gökyüzü elde edebilirsin. Genelde CubeMap kullanılıyor.
* Mobildeki skyboxlar, performans gerekçesiyle daha farklıdır.

## 18-HD Render Pipeline|Universal Render Pipeline

* Render pipeline, grafik işleme hattıdır.
* 3 tür pipeline göreceğiz : Built-in, Universal ve High Definition. Farklı platformlar, farklı seçenekler ve elbette farklı kaliteler için aralarında seçim yapabilirsin. Projeye başlarken seçmen daha doğru, yoksa çok düzeltme gerekir.
* Bunları Package Manager ile çekebilirsin.
* Bunları da (URP ve HDRP) nesne olarak oluşturabilirsin. Böylece grafik ayarları (low,medium,high) seçenekleri oluşturabilirsin. Daha sonra bunları Project Settings/Graphics altından seçebilirsin. Yine Project Settings/Quality’den bunları ayarlara atayabilirsin.
* \*B :: Tüm materyalleri başka bir pipeline’a uygun geliştirmek-değiştirmek için, Edit/Render Pipeline/… Bazı nesneleri yine elle değiştirmen gerekir.
* Render nesnesindeki HDR, yalnız High-Ultra ayarlarda seçilmeli.

# Bölüm-11: Unity Efekt

## 1-Component Efektleri-Trial Renderer

* Trial renderer, kuyruklu yıldız efekti gibi bir şey.

## 2-Line Renderer|Halo|Projector|Lens Flare

* Projector, projeksiyon gibi davranır. Halo yaklaştıkça kaybolur. Lens flare parlama efektidir. Line renderer’ı lazer gibi düşün.

## 3-Script Dosyası İle Efekt Kontrolü

* Objenin child’ını kamera yaparsan, kamera objeyi takip eder.

## 4-Particle System|Particle Effect-I

* Metin2’deki kılıç parlama efektine benziyor. En sık bu particle effect kullanılır.

## 5-Particle System|Particle Effect-II

* Ring Buffer’da en performanslısı ‘Pause…’dır.
* Script tarafında değişimi ve OnParticleSystemStopped() metodunu gördük.

## 6-Particle System|Particle Effect-III

* Particle detaylara devam etti.

## 7-Particle System|Particle Effect-IV

* Particle System’deki ‘External Forces’ı açarsan, efekt rüzgardan etkilenir.

## 8-Particle System|Particle Effect-V

* Particle detaylarına devam ettik.

## 9-Visual Graphic Effect-I

* Particle effect CPU’da, visual effect GPU’da çalışır. Visual effectlerin daha birçok iyi özelliği var.

## 10-Visual Graphic Effect-II

* \*B :: En alt bloğa ‘Orient:Face Camera Position’ eklersen efektin 3 boyutluymuş gibi görünür.
* Update Particle bloğuna ‘Collide with Plane’ eklersen parçacıklar zeminle çarpışır.

## 11-Visual Graphic Effect-III-IV

* Built-in’lerde spark gibi şeyler var. Üzerinde oynamalarla orijinal şeyler yaratabilirsin.
* Triggerları ve diğer özellikleri inceledik.

# Bölüm-12: Unity UI-Canvas-Layout İşlemleri

## 1-Canvas UI Genel Anlatım

* Genel bilgiler.

## 2-Canvas Elemanları-I

* \*Site :: dofont.com
* Canvas elemanlarını elle değil, ‘Rect Transform’ ile hizala.
* Rich Text ile bu textlere HTML gömebiliyorsun.

## 3-Canvas Elemanları-II

* Rascast Target tıklanılabilirliği ayarlar.
* Image’de ‘Image Type’ alanını ‘filled’seçip oradaki ayarlarla oynadık.

## 4-Canvas Elemanları-III

* Button ve Toggle’ı gördük. Toggle’da çift oluşturmak istiyorsan, Toggle Group componentini ekleyebilirsin. İkisine de aynı toggle’ı sürükle.

## 5-Canvas Elemanları-IV

* Slider, Scrollbar, Dropdown
* Dropdown value olarak int döndürür.

## 6-Canvas Elemanları-V

* Input Field işlendi.

## 7-Canvas Elemanları-VI

* Panel ile canvası bölebilirsin.
* Scroll view’da ‘movement type’ genelde elastic(%99 bu) veya clamp seçilir.

## 8-Canvas Elemanları-TextMeshPro Anlatım ve Ayarları

* TMP’ye sprite eklenebiliyor ve bu bir emoji gibi ‘<sprite…>’ ile çağrılabiliyor.

## 9-Layout-Öğe Düzenleme-Layout Çeşitleri ve Ayarları

* Vertical Layout Group ve Layout Element işlendi. İkincisi tek tek düzenlenebilmesini sağlar. Birincisinde de ayarları değiştirerek tek tek düzenleyebiliyorsun. Eğer hem group hem element(tekil) varsa, element daha önceliklidir.
* Grid Layout Group da işlendi.

# Bölüm-13: Unity Animasyon

## 1-Animasyon İşlemleri Anlatım ve Ayarlar-I

* Window/Animation

## 2-Animasyon İşlemleri Anlatım ve Ayarlar-II

* \*B :: Klipler arasında değişiklik yapacaksan, herhangi birinde kullandığın componenti silme. Componenti istemediğin yerde deaktive edebilirsin.

## 3-Animasyon İşlemleri Anlatım ve Ayarlar-III

* Add Event :: Fonksiyon veya olay ekleme.
* Turuncu, default animasyondur. (Animator’de)
* Trigger ile tetikleyici oluşturabilirsin. Boolean ile farkı, boolean değiştirilmedikçe devam eder. Trigger gider geri gelir.

## 4-Animasyon İşlemleri Anlatım ve Ayarlar-IV

* Any State joker gibi davranır. Yalnız çıkış sağlar, kendisine dönülemez.

## 5-Animasyon İşlemleri Anlatım ve Ayarlar-V

* Substate ile animator içinde animator oluşturabilirsin. Dış state’e bunu komple bağlayabileceğin gibi içindeki animasyonları da bağlayabilirsin.

## 6-Animasyon İşlemleri Anlatım ve Ayarlar-VI

* Bu yukarıdakiler animator compenenti. Ayrıca animation olarak tekil bir şekilde de ekleyebilirsin.
* Bağlantılarla (Animator) ilgili ‘Solo’ ve ‘Mute’.
* Animator Inspector’undaki ‘Parameter’ler kod tarafından çekilir. Bu statelerin (Animatordaki hücrelerin) Inspector’u. Buraya da script ekleyebilirsin.

## 7-Animasyon İşlemleri Anlatım ve Ayarlar-VII

* ‘OnAnimatorMove’ güçlü bir fonksiyondur. Animasyonu kesintiye uğratarak istediğini yapar.
* ‘OnStateMove’un eklenebilmesi için ‘OnAnimatorMove’ ana script’te ekli olmalı. ‘OnStateMove’ state script’ine eklenebilir.
* ‘Has Exit Time’ yine state’in Inspector’undaki bir seçenek. Seçiliyse diğer state’e geçmeden ilgili state’in bitmesini bekler.

## 8-Örnek-3D Animasyon Paket-I

* Animator’deki geçişlere orada tuş atayabileceğin gibi scriptte de mevcut geçişlere tuş atayabilirsin. Bunu ‘.Play’ metodu ile yapabilirsin.

## 9-Örnek-3D Animasyon Paket-II

* ‘Has Exit Time’ı kapattı. Böylece duruştan yürüyüşe veya tersine anında geçebildi.
* Zıplama için bool değil trigger kullandı, çünkü eylemin bitmesi gerekiyor. Yani yarım zıplama olmaz.
* State Inspector’da ‘Motion’daki seçeneklerden ‘Root Motion’ seçenekleri yerinde saymaz. Yürür, koşar ve konumu değişir.

## 10-Örnek-2D Animasyon …-Sprite İle Animasyon Yapma

* 2D’de tahmin edebileceğin üzere, sprite’ların art arda oynatılması ile animasyon yapabilirsin. İskelet örneğinde hazır olarak gelmedi. Bu sebeple animator’da sprite’ları art arda koyarak kendimiz yaptık.

# Bölüm-14: Unity Audio-Ses

## 1-Ses Componentleri ve Ses Ayarları-I

* Load In Background bir performans ayarı. Ambisonic ise VR’la ilgili.
* Genelde boyut olarak en fazla sesler yer kaplar.
* Load Type’da eğer ses yüksek boyutluysa ‘Decompress On Load’ kullanma, ‘Compress In Memory’ kullan. Streaming ise kademeli olarak açar.
* ADPCM, Vorbis’e göre daha az bellek kullanır. ADPCM sürekli çalınan (karakter yürüme sesi gibi) kullanılmalı.

## 2-Ses Componentleri ve Ses Ayarları-II

* Kameranda ‘Audio Listener’ componenti yoksa sesleri duyamazsın.
* Audio Source’deki ‘Bypass…’ seçenekleriyle kaynağa diğer ses componentlerinin etkinliğini sağlayabilirsin. Örneğin ‘Audio Reverb Zone’ componenti.

## 3-Ses Componentleri ve Ses Ayarları-III

* Window-Audio Mixer ile objelerindeki ‘Audio Source-Output’ alanına bileşen oluşturabilirsin. Bununla sesleri harmanlayabilirsin.
* Audio Mixer’deki Master, diğerlerinin üstüdür. Yani onu azaltırsan, diğerleri onun üstüne çıkamaz.

# Bölüm-15: Unity AI-Yapay Zeka-Navigation

## 1-Tüm İşlemler-Anlatım ve Ayarlar-I

* Yolu bulacak nesnelere ‘Agent(Ajan)’ denir.
* Window’dan ‘Navigation’u bul. Patikayı oluşturacağın elemanların statik olmalı. Hatta ‘Navigation Static’ var onu seç.
* ‘Agent’daki değerler ‘Bake’dekilerden düşük olmalı. Aksi halde bake edildikten sonra büyük agentler ilerleyemez.
* Agentlara ‘Nav Mesh Agent’ componentini ekle.

## 2-Tüm İşlemler-Anlatım ve Ayarlar-II

* \*G :: Şu an birden fazla agent türü tanımlayabilsen de sadece biri çalışıyor ve bu Unity’nin bir hatası. Düzeltilmiş olabilir.
* ‘Nav-Object’deki ‘Generate OffMeshLinks’i aktive ederek boşluklardan atlamayı sağlayabilirsin.
* ‘Nav-Bake’deki ‘Height Mesh’ merdivenle ilgili bir ayar.

## 3-Tüm İşlemler-Anlatım ve Ayarlar-III

* ‘Ins-Nav Mesh Agent-Obstacle Avoidance’ ile diğer ajanlarla olan çarpışma ilişkilerini hazırlayabilirsin.
* Çok sayıda agent varsa Quality’i azaltmalısın. Çünkü bu külfetli bir işlem.

## 4-Tüm İşlemler-Anlatım ve Ayarlar-IV

* ‘Nav-Areas’da agentler costu düşük yolu daha öncelikli tercih ederler.
* Costu 1 olan yerin devamında 999 olan yer gördüklerinde agentler şaşırır.
* ‘Nav Mesh Obstacle’ componenti ile dinamik engeller yaratabilirsin. Bake edilmesi gerekmez.

## 5-Tüm İşlemler-Anlatım ve Ayarlar-V

* Agentlar için parkur örneği yaptık.

## 6-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-I

* \*B :: Canvası 2D’den çıkarıp 3D öğelere koyabilmek için ‘Render Mode’u ‘World Space’ olarak seçmelisin.
* Pozisyon resetlendiğinde parent’ten miras alınır.

## 7-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-II

Inspector penceresi yanındaki kilide tıklayarak onu sabitleyebilirsin.

## 8-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-III

Quaternion.Identity :: Objenin orijinal rotasyonu.

## 9-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-IV

* \*B :: Agentların trigger’ını açsan dahi trigger’ı açık yerlerde düşmezler.
* Başka bir scripte diğer script’ten GetComponent ile erişebilirsin.

## 10-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-V

* Örneğe devam.

## 11-Yapay Zeka Oyun Örneği-Pekiştirme Dersi-VI

* Normal örneğe bir de Nav. Engeller ekledi. Yani Nav Mesh Obstacle ile uğraştı.

# Bölüm-16: Unity Input-Kontrol Yöntemleri-Klavye-Mouse-Joystick-Touch Dokunma

## 1-Input Manager-İşlemler-I

* Edit-Project Settings-Input Manager
* Input sınıfı ‘UnityEngine’de mevcut, script’e extra using eklemene gerek yok.

## 2-Input Manager-İşlemler-II

* ‘Dead’ analoglardaki(tuş olmayan) input treshold’udur.
* \*B :: Inputları FixedUpdate ile yakala. Bu Update’den daha hızlıdır.
* transform.Translate hareket ettirir. Ama eğer fiziksel kuvvet ile hareket istiyorsan, (Rigidbody değişkeni).AddForce(Vector3) kullan.

## 3-Input Manager-İşlemler-III

* Ders başı input tuş temsilleri :: [1]==Numpad1 gibi.

## 4-Input Manager-Touch İşlemleri-I

* Touch’u gördük. En temel işlemleri ‘phase’ ve ‘postion’.

## 5-Input Manager-Touch İşlemleri-II

* Dokunulan yerin ‘Ray’(Işın) ile konumunu aldık. 2D ve 3D’de farklı alınır.

## 6-Input Manager-Acceleration-Telefon Hareketlerini Yakalama

* İvme ölçeri gördük. ‘dir.x=’ ve ‘dir.z=’ kısımları her zaman kullanılan standart kodlardır.
* Sağ-sol ve ileri-geri hareketleri yakalayabilirsin.

# Bölüm-17: Unity Diğer Çeşitli İşlemler-Çeşitli Yöntemler-İpuçları

## 1-3D-Terrain Anlatım ve İşlemler-I

* Standart Assets’i indirdi storedan.
* Package Manager’dan terrain Tools’u ekledi. Bununla daha gelişmiş işlemleri yapabilirsin. Etkinleşmesi için Unity’i yeniden başlatmalısın.

## 2-3D-Terrain Anlatım ve İşlemler-II

* Terrain’de fırça işlemlerine devam etti.

## 3-3D-Terrain Anlatım ve İşlemler-III

* Fırçalara devam.

## 4-3D-Terrain Anlatım ve İşlemler-IV

* Suyun konumunu ayarlayarak oyukları doldurabilirsin.
* Rüzgar eklediğinde ağaçlar sallanırlar.

## 5-3D-Terrain Anlatım ve İşlemler-V

* \*B :: Terrain Inspector’deki ‘Draw Instanced’ Scene’deyken hesaplanmayıp Game’deyken hesaplanan detayların Scene’deyken de hesaplanmasını sağlar.
* Yine Inspector’daki ‘Base Map Distance’ belirlenen mesafeden sonra çözünürlüğü düşürerek performansa katkı sağlar.

## 6-3D-Terrain Anlatım ve İşlemler-VI

* Inspector’deki ‘Billboard Start’, ‘Fade Lenght’, ‘Max Mesh Trees’; Tree&Detail Objects’in performansla alakalı ayarlarıdır.

## 7-3D-Terrain Anlatım ve İşlemler-VII

* Tüm modeller üçgenlerle oluşturulur. İspatını görüş şeklini ‘WireFrame’ seçerek görebilirsin.

## 8-Past Processing-I

* Oyun sonuna gelindiğinde optimizasyon için yapılır. Package Manager’dan import etmelisin.
* \*B :: En az performans gerektiren ‘Anti-Aliasing(Kenar Yumuşatma)’ yöntemi FXAA’dır. Bu yöntem ışık yoğunluğu fazla olan yerlerde iyi çalışmaz. SMAA ise daha kaliteli sonuçlar verebilir. Ancak %90 FXAA kullanılır. TAA ise hareketli objelerin kenarlarını yumuşatmak için kullanılır.

## 9-Past Processing-II

* Bu işlerde kamera filtrelerine benziyor. Layerlarıyla oynaman gerekiyor kameranın bunu yakalayabilmesi için. Bu sayede bir noktadan başka bir noktaya geçtiğinde farklı efektleri etkinleştirerek oyuna çok daha farklı bir hava katabilirsin.

## 10-3D-2D RayCast Sistemi Anlatım ve İşlemler-I

* Işın göndermede FixedUpdate kullanılır.
* Script’teki RaycastHit, ışının çarptığı objeyle ilgili.
* \*B :: LayerMask ile ışının hangi hedeflerle etkileşime girebileceğini bulabilirsin. Scriptteki 1<<8 ifadesi sadece 8. katmanı dikkate al demek. ‘x=~x’ ile kuralı tersine çevirebilirsin.

## 11-3D-2D RayCast Sistemi Anlatım ve İşlemler-II

* \*G :: Mermi yaparken prefab’tan oluşturup Destroy ile süreli yok etsen de performanstan çok kaybedersin. Bu sebeple Raycast kullanabilirsin. Vurduğun nesnenin Rigidbody’sine erişip ona güç uygulamak iyi bir yol.

## 12-3D-2D RayCast Sistemi Anlatım ve İşlemler-III

* Mermi izi yaptık ve 2D Raycast’ı işledik.

## 13-PlayerPrefs İşlemleri-Load&Save Sisteminin Tüm Detayları ve Örnekleri-I

* PlayerPref ön bellekte tutulur.
* Set ve get işlemleri bittiğinde ‘.Save()’ ile PP’yi kaydetmelisin. Save’in yerinin hiçbir önemi yok. Aslında bu sistem Save olmadan da çalışıyor ancak hata çıkarabilir kullanılmadığı durumda.

## 14-PlayerPrefs İşlemleri-Load&Save Sisteminin Tüm Detayları ve Örnekleri-II

* Ses seviyesi gibi şeyler PP’de tutulur. Örnek olarak slider’dan anlık değişimi yakalayıp Save’ledik.

## 15-Dosya Sistemi İle Save&Load İşlemi-I

* Kaydedilecek veriler static olmalı. Ayrıca ‘SaveSys’de static class olmalı.
* Oluşturulan ‘.bin’ dosyasının konumu projenin dosya konumudur.

## 16-Dosya Sistemi İle Save&Load İşlemi-II

* ‘… as PlayerData’ :: …’dan dönenin türünü PlayerData olarak değiştirir.
* [System.Serializable] :: Sistemde dolaşacak dosyaların serileştirilmesi. Gereklidir.

## 17-2D-Tile Map-Tile Palette-Anlatım ve İşlemleri-I

* Time Map bir grip(Izgara) sistemidir. Tile Palatte’yi yüklemelisin. Yükledikten sonra ‘Window’ sekmesinden ulaşabilirsin.
* Store’dan bir ‘Tile Set’ indirdi.
* Eğer spritelar parçalanmamışsa, ‘Sprite Editor’ ile parçalayarak palette kullanabilirsin.
* Objelerin düşmemesini istiyorsan ‘Tilemap Collider 2D’ eklemelisin.

## 18-2D-Tile Map-Tile Palette-Anlatım ve İşlemleri-II

* Edit’e basarak paletler arasında değişiklikler yapabilirsin.

## 19-Genel İşlem-Unity Event System-Event Trigger-Drag-Click vb. İşlemler

* Event Trigger ile input işlemlerini zenginleştirebilirsin. Bu bir component.
* \*A :: IDragHandler ve IEndDragHandler’i gördük. Bunlar Event Trigger ile ilişkili ve sürükleme olaylarında kullanılırlar. Bununla basit bir item çantası yaptık.
* Bu interfaceler ile Event Trigger’ın farkı, interfaceler ile Event Trigger’daki işlemleri kod tarafında yapabilmen.

## 20-Miscellaneous İşlemler-Çeşitli İşlemler-Video Player

* Render Texture videoyu cisimler üzerine yansıtabilirsin. Bu, Video Player’ın Render Mode’unda bir seçenek.
* ‘Misc.’ Componentler karışık olan, diğer kategorilere giremeyen componentlerdir.
* ‘Aim Constraint’ componentini gördük. ‘Look At Const.’, ‘Position Const.’, ‘Rotation Const.’, ‘Scale Const.’, ‘Parent Const.’ Gördük. Parent Const. çalışması için tahmin edebileceğin üzere akrabalık olması gerek.

## 21-Sahne Yönetim İşlemleri-SceneManager Komutları ve Örnekler

* Build Settings’a tüm sahnelerini eklemelisin. Eğer kod tarafında müdahale etmediysen, ilk sıradaki sahnen menü olmalı.
* LoadSceneMode.Additive’nin ‘.Single’dan farkı, sahneyi açık sahnenin üstüne yüklemesidir.
* Script’teki ‘…+=…’ göreceksin. Bu eğer sol doğruysa sağı çalıştır demektir.

## 22-Unity Standart Asset Paket İncelemesi

İçinde uçak, araba, FPS materyalleri gibi örnekler bulunan bir kaynak. Öğrenmek için kullanabilirsin. Ticari kullanımlarda sıkıntı çıkabilir.

# Bölüm-18: Unity Çeşitli Örnekler-İpuçları

## 1-FPS Anlatım ve Örnekler-I

* Sketchfab’ın tanıtımı.

## 2- FPS Anlatım ve Örnekler-II

* Unity Particle Pack’ı import etti.
* \*A :: Raycastı silahtan gönderirsen ışın yanlış nesneleri vurabilir. Bu sebeple ışını kameradan göndermelisin.

## 3-Sahne Geçiş-Loading Ekranı Yapımı-Slider Ayarları

* Sadece scripti paylaştı. Mantığı kolay. Yalnız ‘Coroutine()’ kullanımını tam hatırlamıyorum.

## 4-ScrollView Örneği-Oyunlardaki Marketlerde Kullanılan Sistem

* Hem vertical hem horizontal group ekleyemezsin. Aslında bunun yerine ‘Grid Layout Group’ ekleyebilirsin.

## 5-Object Pooling-Obje Havuzu Yöntemi

* Objeleri sürekli oluşturup yok etmek yerine bir havuz oluşturup sürekli aynı objeleri kullanırız. Çünkü oluşturmak ve yok etmek CPU’ya külfet. Bu yöntem basit bir dizi ile gerçekleştirilir.

## 6-Ragdoll Sistemi Nedir-Ragdoll Oluşturulması

* Char Joint’in otomatik oluşturulması. Tavsiye edilen, bu oluşturmadan sonra ince ayarları elle yapmaktır.
* Animasyon açıkken ragdoll aktif değildir. Bunu kapattığında bir ölüm efekti elde etmiş olursun ragdoll ile.

## 7-Oyunlardaki Mini Map-Anlatım ve Yapımı

* Kameralarda önceliği ‘Depth’ ile ayarlarsın.
* A\* :: Mini Map kameran Orthographic olmalı. Bu görüntüyü Canvas içinde RawImage’ye yansıtırsın. Buradaki detaylar bir tık havada kaldı.
* Raw Image’ye maskeleme yaparak haritaya çerçeve koyabilirsin.

## 8-Way Point-Anlatım ve Örneği

* Oto ilerleme sistemini yaptık.

## 9-Oyunlardaki El Feneri sistemi

* Spotlight kullandık. Cookie vererek daha gerçek görünmesini sağladık.

## 10-Oyunlardaki Tab Menü Sistemi

* Basit bir Options menüsü yaptık.

## 11-Oyunlardaki Build(İnşa) Yapma-Obje Oluşturma Sistemi-I

* Simcity gibi oyuncuya inşa hakkı veriyoruz.
* Plane’e bir katman atadı ve Ray’i buna gönderdi. Aksi halde bir eksen sorun yaratır ve nesne konumlanamaz. Koddaki (1<<8) bunu yapıyor.
* …=new Vector3(…,hit.point.y+.5f,…) olmazsa objenin yarısı aşağıda kalır. Bu pivot noktasıyla alakalı bir sorun.

## 12-Oyunlardaki Build(İnşa) Yapma-Obje Oluşturma Sistemi-II

* Build’de üst üste binmeleri engelledi.

## 13-Oyunlardaki Destruct(Obje Yıkma-Parçalama) İşleminin Yapılması-I

* Parçalama için nesne buna uygun olarak modellenmiş olmalı.
* Tüm parçaların Mesh Collider-Convex’ini açtı ve RigidBody ekledi.
* İlk yöntemde mouse’a tıkladığında normal obje yerine kırılabilecek objenin geçmesini sağladı. Yani biraz hileye başvurduk.
* Ray’in çarpması için Collider ve RigitBody lazım mı?(Soru-6)

## 14-Oyunlardaki Destruct(Obje Yıkma-Parçalama) İşleminin Yapılması-II

* İkinci yöntem ise patlamadan önce parçaların Rigidbody’sini kaldırıp patlamada vermek. Bu diğer yöntemden daha performanslı.

## 15-Oyunlardaki Kumaş&Bayrak Efekti-Cloth Component-I

* \*G :: Cloth Componenti’i ekledi. Eklenen objeden meshi kaldırmalısın.

## 16- Oyunlardaki Kumaş&Bayrak Efekti-Cloth Component-II

* Kumaş nokta ayarlarını ‘Scene’ ekranında yapmalısın.
* Havlu tarzı bir şey yapacaksan ‘Self Collision’ ayarlamalısın. Yoksa kendisiyle iç içe girer.

## 17-Oyunlardaki Oyun İçi Grafik Ayarının Yapılabilmesi

* \*A :: PlayerPrefs kullanmadan önce Edit altından eskilerini temizleyebilirsin.

# Bölüm-19: Multiplayer Sistemi-Online Oyun-Server İletişimi-Photon Sistemi

## 1-Multiplayer Sistemi-Anlatım ve Örnekler-I

* Asset Store’dan PUN2’yi import etti. PUN, yazılımsal destek de içeriyor. Bu sebeple düz sunucu kiralamaktan pahalıya geliyor.

## 2-Multiplayer Sistemi-Anlatım ve Örnekler-II

* Sıralama, önce sunucuya, sonra lobiye, en son odaya bağlanır.
* Oda kurmadan odaya giremezsin.
* Photon View ekli obje diğer kullanıcılara görünür. Tabi içinin de dolması gerekir. Bunun Animator, RB gibi birçok türü var.
* Sunucuya gönderilecek objede ‘Resources’ klasörüne bağlantı olması gerekmekte.
* ‘oyuncu.cs’ içindeki pw.IsMine olmazsa kullanıcı hareketleri karışıyor. Bununla ayırıyoruz.
* Eksik component vb. sebeplerle senkronizasyon sağlanamazsa, diğer componentler dahi görülemeyebilir ve sonucu olarak oyuncular birbirlerini duruyor gibi görür

## 3-Multiplayer Sistemi-Anlatım ve Örnekler-III

* /

## 4-Multiplayer Sistemi-Anlatım ve Örnekler-IV-V

# Bölüm-20: Unity Çeşitli Pencerelerin Yönetimi

## 1-Project Settings-Audio-Editor-Unity Remote Sistemi-Remote 5

* Unity remote’u telefona indirerek build almadan telde game’i açabilirsin.

## 2-Project Settings-Graphics-Physics 3D-Physics 2D

* \*A :: Ne kadar kaliteli texture kullanırsan kullan, buradaki ayarları doğru yapamazsan istediğin sonucu alamazsın.

## 3-Project Settings-Player-PC&MAC&LINUX Standalone-I

* Resolution’daki ‘Fullscreen Mode’daki 2 seçenekten biri sadece Windows’a biri ise sadece MAC’a özeldir.

## 4-Project Settings-Player-PC&MAC&LINUX Standalone-II

* En başta çıkan Unity ekranını da ‘Player’ sekmesindeki ‘Splash Image’den ayarlayabilirsin.
* Dynamic Batching normalde seçili değil, onu açabilirsin.

## 5-Project Settings-Player-Android

## 6- Project Settings-IOS-WebGL-Universal Windows

## 7-Project Settings-Preset Manager-Quality

## 8-Preferences Penceresi

## 9-Unity Services Penceresi

# Bölüm-21: Unity Optimizasyon

## 1-Script Optimizasyon İpuçları

* Tüm tüyolar scriptte.

## 2-Profiller Penceresi-Genel Optimizasyon Konuları

* Hareket etmeyen nesnelerin static olmalı.
* \*A :: Eğer çarpışma yakalamayacaksan box collider’ı kaldır.

# Bölüm-22: Unity Reklam-Oyun İçi Satın Alma Sistemi-Build İşlemleri

## 1-2023 | OYUNLARA REKLAM EKLEME | ADMOB | PAKET KURULUMU | UYGULAMA OLUŞTURMA

* AdMobs linki bu videoda. Reklam eklemek için AdMobs hesabı gerekli.
* Test reklam kimliklerinin linki bu videoda.
* Android paketleri (SDK-NDK) yüklüyse hata almazsın.
* Konsolda IOS hatası verecek Android geliştiriyorsan. İlgili pathe giderek onları silebilirsin. Sonra Assets/External Dep. Man./Android Resolver/Force Resolver tıkla. Bunu yaptığında entegrasyon tamamlanmış olacak.
* Anlatımda farklı scriptler üzerinden ileliyor ancak gerçekte tek bir script ile halledebiliriz reklam işini.
* Ekledikten sonra “Reklam birimleri”ne bas ve türleri (reklam türlerini) incele.
* Reklam birimi oluşturduğunda 2 tane kod gelecek. Biri uygulama kimliği biri reklam birimi kimliği. Uygulama kimliği değişmez.
* \*G:: Test sürecinde gerçek reklam kimliğini kullanırsan banlanırsın.
* \*B:: Oyun geliştirdin ve artık yayına çıkacaksın. O zaman uygulama kimliğini kopyala ve Assets/Google Mob Ads/Settings a git ve ID’yi yapıştır.
* Settings’deki tikleri seçmen önerilir.

## 2-2023 | OYUNLARA REKLAM EKLEME | ADMOB | GEÇİŞ REKLAMI

* Scriptteki #if UNITY\_EDITOR altındaki Android için, #elif UNITY\_IPHONE ise IOS için olan reklam kimliğidir. Bunun avantajı tek scriptte hem Android hem de IOS için reklam olayını halletmektir.
* Interstitial Ad==Geçiş Reklamı
* Start içindeki “MobileAds.Initialize(….) yalnızca bir kez çalıştırılmalı. Bu SDK’yı başlatır. Aksi takdirde hata alabilirsin.
* Script’teki “GecisReklamiOlustur” fonksiyonu içindeki if(….!=null) standattır. Değişkenin içi doluysa orada temizler.
* “GecisReklamiGoster()” fonksiyonu içindeki if(…&&….CanShowAd()), reklamın gösterilebilir olup olmadığını kontrol eder.
* Reklamlar tek kullanımlıktır. Bu yüzden reklam gösterildikten sonra, gösterilen reklamın durumuna göre yeni aksiyonlar almalısın. Yeniden serverden reklam çekmelisin. Reklam kapatılması vb. olaylarda alınacak aksiyon fonksiyonun “ReklamOlaylariniDinle()”. Burada dinleyiciler(Event Handler) kullanılıyor.
* Reklam dinleyiciyi, oluşturma fonksiyonunun altında çağırarak çalıştırırsın. Dinleyiciyi buna abone ediyorsun.
* Dinleyicide kapanma ve fail durumlarında oluşturucuyu tekrar çağırırsın. Böylece yeniden yükleme sorunu çözülür.
* \*B:: “+=” işareti, bir olaya abone olmayı ifade eder. Yani olay tetiklendiğinde bu kod bloğunu çalıştırmak için kendimizi olayın “abonesi” yapıyoruz.

## 3-2023 | OYUNLARA REKLAM EKLEME | ADMOB | ÖDÜLLÜ REKLAM | ÖDÜLLÜ GECİŞ REKLAMI

* Ödüllü reklamın geçiş reklamından(bir önceki tür) farklı olan kısmı “ReklamGoster” kısmı. Bu fonksiyondaki “.Show” kısmında, AdMob’da girdiğimiz ödül türü ve miktarını çekebiliyoruz.
* Ödüllü reklam kullanıcı rızasıyla gösterilir. Ödüllü geçiş ise belli bir saniyesi sonunda ödül verilir? Yani ödüllüde rıza söz konusu.

## 4-2023 | OYUNLARA REKLAM EKLEME | ADMOB | BANNER REKLAMI

* Banner reklam türü, diğerlerinden farklı çünkü sabit bir yerde ve etkileşim olmadan var olacaktır.
* \*G:: Banner’ı kullanıcının hata ile dokunacağı yere özellikle koymak ban sebebidir.
* Banner boyutu ayarlama=> dk 8:50

## 5-OYUN İÇİN SATIN ALMA EKLEME İŞLEMİ | GERÇEK UYGULAMA YÜKLEME VE TEST İŞLEMLERİ 1

## 6-OYUN İÇİN SATIN ALMA EKLEME İŞLEMİ | GERÇEK UYGULAMA YÜKLEME VE TEST İŞLEMLERİ 2

## 7-BUİLD ALMA | OYUNU PAKETLEME | GOOGLE PLAY | ANROİD

## 8-GOOGLE PLAY'E UYGULAMA YÜKLEME ANLATIM | ÜRETİM | BETA | ALPHA KANALLARI

## 9-BUİLD ALMA | OYUNU PAKETLEME | PC | MAC | WEBGL

## 10-BUİLD ALMA | OYUNU PAKETLEME | İOS

# Bölüm-23: Ücretsiz Assets Bulabileceğin Web Siteleri

## 1-Genel Anlatım ve Açıklama-Genel Asset-Model-Ses-Texture

* LOD sistemi baya avantajlı. (Soru-7)
* Sketchfab’da free modelleri koyarken atıf vermen gerek, aksi sıkıntı.

# Bölüm-24: Örnek Oyun-3D-PC-FPS-Mahalleni Koru

## 1-Mahalleni Koru 1 - Proje Oluşturma | Materyalleri ekleme

* “Low Poly Street Pack” ve “Town Houses Pack”ı Asset Store’dan indirdi.
* Silah modellerini sketchfab.com dan indirdi. İndirdiklerini ticari olarak kullanabilirsin ancak yapanın adını oyunda belirtmelisin.
* Başlamadan yarattığı klasörler: Animasyon, images, Karakter, Material, Script, Sesler, Silahlar.
* FPS için silahı tutacak olan eli indirdi. Video kaynağında var.
* Terrain genellikle ağaç, dağ, tepe oluşturacaksan kullanılır. O yüzden Plane kullandık.

## 2-Mahalleni Koru 2 - Sahnenin ve Mahallemizin oluşturulması

* Eklediği yolların boyutunu karşılaştırmak için “Standart Assets”i import etti(Paket dahili download yok). GUIText hatası verirse koddaki GUIText’leri kaldır.
* Standart Assets’teki Characters/FirstPerson…/Prefabs/FPSController’ı kullandı.
* Material klasörü altına textures klasörü açıp indirdiği texture’leri oraya attı. Sonra “Duvar” materyali oluşturup onu doldurdu. Materyalin “Emission” altındaki Tilling değeriyle oynayarak tuğla boyutlarını ayarladı.
* Teknik objeler altına “DusmanDogmaNoktasi” ve “DusmaninZararVerecegiNesne” empytlerini yarattı.
* Harita karanlıktı ve bunu “Lightning” altındaki “Generate Lightning” ile çözdü. Daha karanlık olan bölgeler için “Reflection Probe” ile aydınlatma yaptı.
* Collider sınırlarını çok iyi ayarlaman gerekiyor. Yoksa karakter oradan geçmese de glitch’i görür.

## 3-Mahalleni Koru 3 - Karakterimizi ekleme | Crosshair | Ateş etme | Ses | Efekt

* Sesleri hem Asset Store’dan hem de freesound.org dan bulabilirsin.
* Raycast’ı kameradan çıkarttık. Daha önce görmüştük.
* Silah sesinin “Audio Source” componentinin “Play On Awake”ini kapattık. Yoksa oyun başlar başlamaz ateş sesi verir.
* Menzili ayarladı. Ayrıca AtesEt fonksiyonundaki ses ve efekt başlatmayı if’in dışına aldı. Böylece if çalışmasa bile (herhangi bir hedef vurulmasa bile) ses ve efekt çalışacak.
* Yalnızca sahnede kullandığı dosyaları bırakmış indirilebilire. Geri kalanları silmiş.

## 4-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 1.KAYNAK DOSYASI

## 5-Mahalleni Koru 4 – Mermi izi | Kan efekti | Vurulan objeye fizik uygulama

* Mermi izi ve kan efekti yapıyoruz. Efektleri AtesEt() fonksiyonu içinde Instantiate() metodu ile oluşturacağız. 3’üncü argümanı açıyı ayarlar Quaternion ile. Buradaki kan ve mermi efektlerinin de standart assetten aldı.
* \*B:: Açı ayarlamadan kasıt, vurulan objede oluşan efektin bize bakmasını sağlaması.
* Mermiyle eşya devirmede en sağlıklı yol, AddForce()’a (-hit.normal) vermek. Böylece vurulan yönün tersine kuvvet uygulanacak.

## 6-Mahalleni Koru 5 - Silah Paketi yükleme | Animasyon Yapma

* Burada kullandığı silahlar video altında mevcut.
* \*G:: Materyalin metalic kanalına hem metalic hem de Roughness materyal atabilirsin.
* Silah animasyonu yaptı bu videoda.

## 7-Mahalleni Koru 6 - Animasyon Yapımı Devam

* Şarjör sesini animasyonda event ile bağladı. Event ile bir fonksiyonu istediğin zaman animasyon içinde çağırabiliyorsun.

## 8-Mahalleni Koru 7 - Şarjör ve Mermi Sistemi yapımı

* [Header(“”)] kullandı. Bu scriptteki public değişkenlere editörde başlık ekler. Daha derli toplu olur.
* TMP’da “Autosize” açarsan, text belirlediğin sınırlara uymak için otomatik büyür ve küçülür.

## 9-Mahalleni Koru 8 - Şarjör ve Mermi Sistemi yapımı 2

* Şarjör değiştirme algoritmasını tasarladı.
* IEnumerator ile önce animasyonu oynatıp, o bitene kadar (.5f) bekletip mermi sayısını süre sonunda güncelledi.

## 10-Mahalleni Koru 9 - Boş Mermi Kovanı Çıkarma | Yeni Kaynak ekleme

* Boş kovan için ayrı bir script açtık. Çünkü silahtan ayrı olarak ses efektleri, yere düşme sesi, çıkış yönü vb. düşünülmesi gereken birçok ayar var.
* Mermi izi, mermi gibi üretilip yok edilen nesneleri klasik yol ile oluşturup yok etmek iyi bir seçenek değil. Daha önce gördüğümüz obje havuzu yöntemini kullan.
* \*G:: AddForce() yerine AddRelativeForce() kullanımını önerdi. Bunların farkı ::
* **AddForce()**: Bu fonksiyon, dünya koordinatlarına göre bir kuvvet uygulamanızı sağlar. Yani, nesnenin hareket yönü ve rotasyonu göz ardı edilir. Eğer nesne yavaşça hareket ediyorsa veya rotasyonu düzgün değilse, kuvvetin etkisi nesnenin davranışını karmaşık hale getirebilir.
* **AddRelativeForce()**: Bu fonksiyon ise nesnenin kendi yerel koordinatlarına göre bir kuvvet uygulamanızı sağlar. Yani, nesnenin rotasyonuna bağlı olarak kuvvetin yönü değişir. Bu genellikle daha öngörülebilir sonuçlar doğurabilir, çünkü nesne kendi koordinatlarına göre hareket eder.

## 11-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 1 | Mermi Kutusu 1

* Mermi kutusunu almayı hem kutuya Raycast atarak hem de collider’dan yakalayarak tasarlayabilirsin.
* Mermi kutusunun detaylarını kendindeki bir script’ten hallettik. Hem böylesi daha doğal hem de silahların script’ine gömmek orada karmaşaya sebep olacak.

## 12-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 1 | Mermi Kutusu 2

* Mermi kutusunun çıkış lokasyonunu “GameKontrol” üzerinden yaptık. Ama GameKontrol’de genelde daha genel işler yapılır. O sebeple yine GameKontrol’de ayrı bir script oluşturarak kontrol sağlayabilirsin, videoda da böyle yaptı.
* GameKontrol script’i genelde kanvastaki işlemleri ve genele yönelik işlevleri kontrol için kullanılır.

## 13-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 1 | Mermi Kutusu 3

* Daha önce karşılaştığın objeye animasyonu verince oluştururken pozisyonunu ayarlayamama ile burada da karşılaştık. Çözümü animasyon objesini bir boş objenin altında yapmak. Yani objeyi boş obje altına almak.
* Oyunu başlatmışken Pause’a basarak Scene’de kaldığın yere bakabilirsin.

## 14-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 1 | Mermi Kutusu 4

* Burada alındıktan 5 saniye sonra rastgele noktadan çıkan tek bir mermi kutusunun scriptini yazdı. Sonraki videoda farklı bir yol izleyecek. Sonraki videoda farklı bi yol izleyecek. Bu video altındaki şu anki script mevcut.

## 15-Mahalleni Koru 10 - Yerden mermi alma sistemi | Yöntem 2 | Mermi Kutusu 5

* Bu sefer her 5 noktada da birer adet çıkabilecek.
* İf-else içinde de continue yazarak döngüde başa dönebilirsin.

## 16-Mahalleni Koru 11 - PlayerPref sistemini kurma | Mermi kaydetme

* Kalan mermiyi PlayerPrefs ile kaydedip sonraki oturumlara saklamak için bir algoritma oluşturdu.

## 17-Mahalleni Koru 12 - (İsteğe Bağlı Part 1)Diğer silahların eklenmesi ve Animasyon

* Pompalının reload ve ateş etme animasyonları yapıldı. Reload’da baya bir emek sarf etti.

## 18-Mahalleni Koru 13 - (İsteğe Bağlı Part 2)Diğer silahların eklenmesi ve Animasyon

* Sniper’ın tutuş ve animasyonlarını yaptı.

## 19-Mahalleni Koru 14 - (İsteğe Bağlı Part 3)Diğer silahların eklenmesi ve Animasyon

* Magnum tutuş ve animasyonlarını yaptı.

## 20-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 2.KAYNAK DOSYASI

## 21-Mahalleni Koru 15 - Sniper Silahı Scope (Yakınlaştırma) işlemini yapma

* Animatörde parametre kullandı. Bu parametreyi script tarafında Mouse 1’in up-down olmasına göre true-false yaparak scope’un aktifliğini ayarladı.
* Zoom (Scope) yapıldığında silahın görünmemesi için Kamera’nın “Culling Mask” özelliğini kullandı. Gizlemek istediklerine yeni bir layer tanımlayarak bunu gerçekleştirdi.
* Culling Mask’ın script tarafındaki kullanımı biraz değişik. Yine inceleyebilirsin. Mesela bunu temel duruma geri döndermek için “Culling\_mask=-1” yazdı.

## 22-Mahalleni Koru 16-AK47 | Magnum | Pompali | Scope (Yakınlaştırma) işlemini yapma

* Ak47’de zoom’dayken ateş edince normal ateş etmedeki animasyon oynuyordu. Bunu kod tarafında, bool değişkenlerle zoom’dayken animasyonu oynatmayarak çözdü.
* Pompalı ve magnumu da aynı şekilde halletti.

## 23-Mahalleni Koru 17 Kod Revize işlemleri | Camera Shake (Kamera Titremesi) yapımı

* Animasyon tamamen bittiğinde mermilerin düzenlenmesini yaptı. Her silah için revize etti.
* Kamera titremesini yaptı.

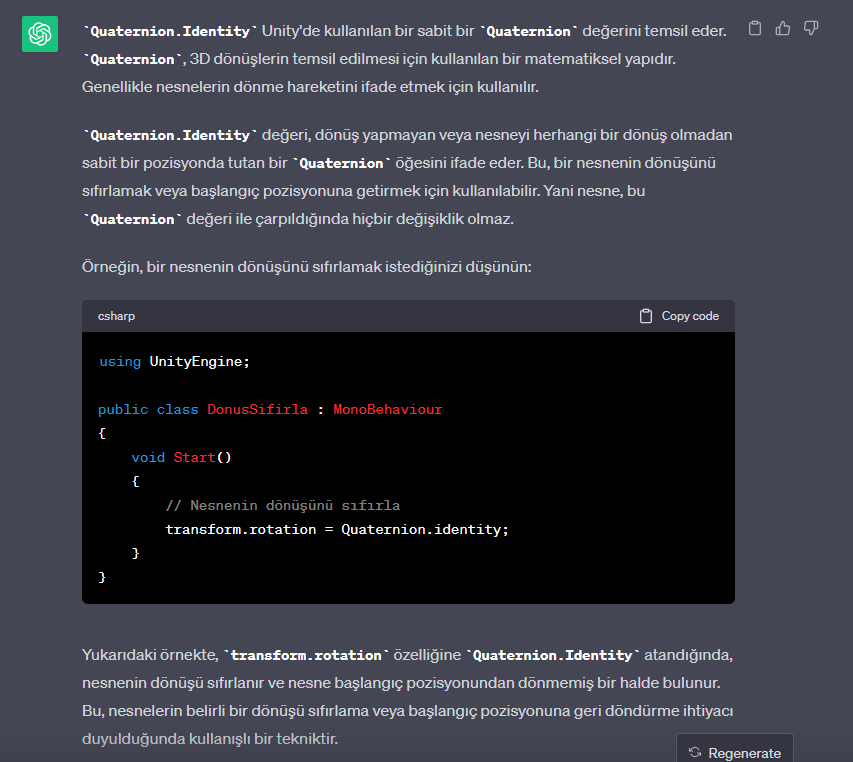
## 24-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 3.KAYNAK DOSYASI

## 25-Mahalleni Koru 18 - Silah değiştirebilme sisteminin yapılması

* Alpha1 klavye üstü 1’dir.
* Hem Q ile hem de klavye üstü numaralarla silah değiştirmeyi yaptık.

## 26-Mahalleni Koru 19 - Yapay Zekanın oluşturulması | Düşmanların İşlemleri 1

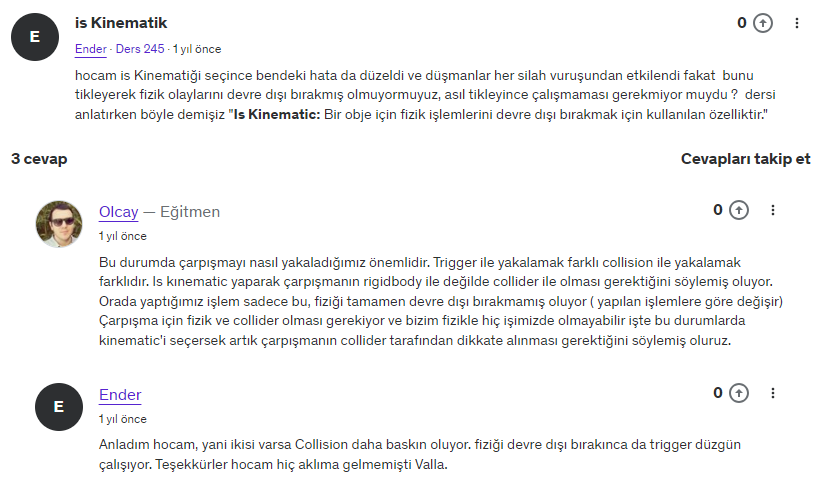
* Yolları “Navigation Static” türüne çevirdi. Yol üstü cisimleri de çevirdi.
* Quaternion.Identity() ::



## 27-Mahalleni Koru 20 - Yapay Zekanın oluşturulması | Düşmanların İşlemleri 2

* Ajanlar artık hasar alıyor. Ayrıca hedef noktalarına ulaşabildiklerinde hasar da verebiliyorlar.

## 28-Mahalleni Koru 21 - Yapay Zekanın oluşturulması | Düşmanların İşlemleri 3

* Burada hedefleri Raycast ile vurmak için IsKinematic’i kapattı. Nedeni:: 

## 29-Mahalleni Koru 22 - Yerden Sağlık alabilme işleminin yapılması

* Burada Healt noktalarını oluşturdu ve can alma sistemini tasarladı.

## 30-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 4.KAYNAK DOSYASI

## 31-Mahalleni Koru 23 - Toplam Düşman Sayısı işlemini yapma | SKYBOX EKLEME | DİĞER

* Düşman sayısını ayarladı ve kalan düşmanı UI’a yazdırdı. Ayrıca düşmanlar bitince kazandın panelini ekledi.

## 32-Mahalleni Koru 24 - Düşman karakterlerini ekleme

* Karakter sitesinden aldığı karakterleri ekledi.
* Sonra “Character Controller” komponentini bu karakterlere ekledi. Bu aynı zamanda Collider görevi de görüyor.
* Ayrıca bu karakterler Scene’de yarısına kadar yere batmış görünürken Game’e geçtiğinde normal görünüyorlar.

## 33-Mahalleni Koru 25 - Düşman Animasyonlarının Yapılması | Koşma ve Ölme

* Zombi yürüme animasyonunu indirdikten sonra “Rig” kısmından “Animation Type”ı Generic’ten Humanoid’e çevirdi. Ayrıca “Loop Time”ı da tikledi.
* Zombiye animatör ekledi hareketi yakalayabilmek için. Ayrıca “Culling Mode’u “Cull Update Transforms” seçti. (???)
* Ölme animasyonunda karakter yine yerden yukarıda öldü ve bunu değişik bir yolla çözdü. Dakika 30 civarı.
* Animatörde koşma ve ölmeyi okla bağladı ve ölmeyi SetTrigger() ile tetikledi. Sonra ince ayar yaptı ve geçiş oldukça yumuşak oldu.

## 34-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 5.KAYNAK DOSYASI

## 35-Mahalleni Koru 26 - Genel İyileştirmeler | Zombilere ve Oyun içine Ses ekleme

* Birkaç düzeltme yaptı ve oyun müziği ile zombi seslerini ekledi.

## 36-Mahalleni Koru 27 - Bomba Atabilme özelliğinin Entegresi 1

* Bombanın atılma açısını karışık bir komutla ayarladı.
* Bombanın patlama alanındaki tüm Collider’ları yakalamak için “Physics.OverlapSphere(…)” kullandı. Bununla yakalananları Collider[] dizisine çekti.
* Bombanın patlama algoritması ve kendisine özgü güç fonksiyonu (“AddExplosionForce”) gibi detayları var.

## 37-Mahalleni Koru 28 - Bomba Atabilme özelliğinin Entegresi 2

* Patlama algoritmasında çarpışmada bombayı patlayacak şekilde dizayn etti ancak bu baya sorun yarattı. Çünkü bomba ilk kez çarptıktan belirli bir süre sonra yok oluyor ve birkaç kez çarptığında çarpmaya bağlı efekt ve ses birçok kez oynatılıyor. Bunu ses ve efekti bombaya bağlamak yerine bomba script’inde Instantiate ederek çözdü. Daha sonra obje kalabalığı olmaması için ses ve efekte kendi scriptlerini oluşturarak belirli bir sürede yok etti.

## 38-Mahalleni Koru 29 - Item (Sağlık ve Bomba) Toplama ve Kullanma sistemi 1

* Sağlık kutusunu alınabilir ve biriktirebilir hale çevirdi.

## 39-Mahalleni Koru 30 - Item (Sağlık ve Bomba) Toplama ve Kullanma sistemi 2

* Bomba kutusu da alınabilir ve biriktirilebilir hale çevrildi.

## 40-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 6.KAYNAK DOSYASI

## 41-Mahalleni Koru 31 - Çeşitli Ayarlar & Düzenlemeler

* Hareket etmeyecek efektleri de statik yapabilirsin
* Performansı tüketen şeyler ışık ve gölgelerdir. O yüzden trafik ışıklarının lambalarını kapatabilirsin (pasif hale getirebilirsin) veya Baked yapabilirsin.
* Ana obje dinamik olsa da çocukların hareket etmeyecek (ana objeye göre hareket etmeyecek) bağlı parçalarını statik yapabilirsin.

## 42-Mahalleni Koru 32 - Çeşitli Ayarlar & Düzenlemeler

* Occlusion Culling uyguladı. Bu, görmediğin nesnelerin hesaplanmamasını sağlar. Kameranın Occlusion Culling komponenti açık olmalı.
* Post-Process ile kamera filtresi uyguladı.

## 43-Mahalleni Koru 33 - Mermi Çıkma ve gitme işleminin yapılması

* Mermi oluşturdu, buna ileriye doğru güç uyguladı ve “Trail Renderer” ekleyerek merminin arkasında iz bırakmasını sağladı.
* Merminin yok olmasını hem OnTrigger ile hem de Start içinde süreli Destroy() metoduyla sağladı çünkü mermi hiçbir collider’a çarpmayabilir.

## 44-Mahalleni Koru 34 - Bazı işlemlerin geliştirilmesi

* Zombi öldükten sonra tagını “Untagged” olarak değiştirdi çünkü ateş etmeye devam ettikçe ölme fonksiyonu çalışıyordu ve bu hataya neden oluyordu.

## 45-Mahalleni Koru 35 - Oyun sonu panellerinin değiştirilmesi

* Pause menüyü tasarladı ve ana menü için hazırlık yaptı.

## 46-Mahalleni Koru 36 - Çeşitli iyileştirmelerin ve düzeltmelerin yapılması

* Oyunu durdurduğunda scriptler açık kalıyor, bu yüzden hareket edemesek de kamerayı oynatabiliyoruz. Çözümü durumlara göre scriptleri aktif-pasif yapmak.
* Menüde cursor görünmüyordu. Bunu “Cursor.visible=true” ve “Cursor.lockState=CursorLockMode.None” ile çözdü.
* Bir scripte başka bir scriptten erişirken erişeceğin scriptin eğer varsa namespace’i erişeceğin scriptte using ile ekli olmalı.
* FirstPersonShooter scriptindeki alanlar Serialized olduğu için private olsalar da dışarıdan erişilebilirler. Bu scriptteki Lock işlemini değiştirebilmek için scriptteki ilgili alana giderek onu public yaptı.

## 47-Mahalleni Koru 37 - Ana Menü Yapımı

* Ağaçlar statik olduğundan yerini değiştirdiğinde Game ekranında görünmeleri için bake yapman gerek.

## 48-Mahalleni Koru 38 - Loading ve Oyundan çıkış panel yapımı

* Kodları IEnumerator içine aldı ve loading slideri için yazdığı while döngüsü içine “yield return null” ekledi. Bu, while döngüsünün her framede bir kez çalışmasını sağlar.

## 49-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 7.KAYNAK DOSYASI + BUİLD ALINMIŞ HALİ

# Bölüm-25: Örnek Oyun-2D-Mobil-Multi Player Online Oyun-Kutuma Dokunma

## 1-Kutuma Dokunma-I

* ‘Pixel Art Platformer-Village Props’ kullandı.
* Sprite Editor ile parçaladı. Windows/2D’den Tile Paleti açtı ve sahneye bir Tilemap ekledi. Bunu zemin için yaptı.

## 2-Kutuma Dokunma-II

* Sahneyi dizayn etti.

## 3-Kutuma Dokunma-III

* Topa Polygon Collider 2D’ ekledi. Bu şekli otomatik sarmalar.
* Rigidbody’nin AddForce fonksiyonundaki ‘ForceMode2D.Impulse’ sürekli değil tek seferlik bir güç uygulanmasını sağlar. Bunu gülleye ekledik.

## 4-Kutuma Dokunma-IV

* Çarpışmaları yakalamak için kutuların IsTrigger’ını açamazsın, düşerler. Topunkini açtık, bu sefer de top içlerinden geçecek. O sebeple çarptığı an IsTrigger’ı kapatarak sorunu çözdük. Sonra bundan vazgeçti, gülle çarpınca yok etti.
* Healtbar’ın belirli bir süre görünmesi için, ‘IEnumerator CanvasÇıkar()’ı yazdı ve bunu StartCoroutine(…) ile çağırdı.

## 5-Kutuma Dokunma-V

* ‘Carton FX Free’, ’48 Particle Effect Pack’ ve ‘Standart Assets’ten particle effectler bulabilirsin.
* ‘TopAtisEfekt’, Instantiate ile oluşturdu ve böylece konumu direkt oradan verdi ancak sen bunu farklı şekillerde yapabilirsin.

## 6-Kutuma Dokunma-VI

* Gülle çarptığında yok oluyor ancak sesi ve efekti de yanında götürüyor. Bu sebeple bunları ‘GameKontrol’e taşıdık.

## 7-Kutuma Dokunma-VII

* Canvas Scaler’dan ‘Scale With Scene’ seçmeyi unutma.

## 8-Kutuma Dokunma-VIII

* \*G :: İşin içine sonrada server gireceğinden, her şeyi GameKontrol’de toplamaya çalışıyor. Yoksa oyuncu sağlığı, oyuncu nesnesinde de tutulabilir.
* AddForce’u Oyuncu2 için negatif olarak değiştirmelisin, yoksa kendini vurur.

## 9-Kaynak Dosya-I

## 10-Kutuma Dokunma-IX

* Powerbar’ı yaptık.
* Coroutine’in çalışıp durmasını değişkenle kontrol etme. ‘StopCoroutine’ kullan. Aksi halde zamanlama şaşar.

## 11-Kutuma Dokunma-X

* Buradan sonra sunucu işin içine girecek.
* Panelle ilgili ayarlamaları yaptık.

## 12-Kutuma Dokunma-XI

* PUN’u kullanmak için oyuna import etmelisin.

## 13-Kutuma Dokunma-XII

* OlusacakNokta\_1 ve …\_2 pozisyon olmak için var. Bunlar, Oyuncuların yalnızca transformunu taşıyan nesneler.
* Serverde oluşacak nesneler için ayrı bir klasör açmalısın. Bunun adı ‘Resources’.
* SunucuYonetimi scripti sahneler arasında yok olmamalı. Bu sebeple ‘DontDestroyOnLoad(…)’ı ekledik.

## 14-Kutuma Dokunma-XIII

* Oyunu sunucuya uyarlamaya devam ettik.

## 15-Kutuma Dokunma-XIV

* Topu(gülleyi) oyuncuya spesifik yaptığımızdan, Powerbar’da sıkıntı çıktı. Bu sebeple, Top.cs’ye oyuncu tagını çektik ve buna göre işlem yaparak sorunu çözdük.
* Sahnede hareket edebilecek veya değişecek her şeyi sunucu tarafına taşımamız gerekiyor.

## 16-Kutuma Dokunma-XV

* \*A :: Resources’dekiyle aynı isimde PreFab bulunmasını Photon istemiyor.
* [PunRPC] varsa metodun üstünde, o metoda diğer scriptler erişebilir. Erişeceğin yerde bunu ‘.RPC’ ile çağırırsın.
* Tag belirlemeyi de sunucu yönetimine verdik.

## 17-Kutuma Dokunma-XVI

* GameKontrol scriptini prefab(yani Resources’e ekleyebilirdik) yaparak da devam edebilirdik ancak bunun yerine içindekileri Resources’deki elemanlara paylaştırmaya karar verdik.
* \*G :: Dumanpuf efektinin scriptine gidip ‘Destroy’ metodunu ‘PhotonNetwork Destroy’ olarak değiştirdik. En sonda Photon’daki garip bir hata ile karşılaştık ve bunu ‘PhotonNetwork.Destroy’u if(pw.IsMine) içine alarak çözdük.

## 18-Kutuma Dokunma-XVII

* Resources’deki ‘Oyuncu’ prefabının tagını yeniden Oyuncu\_1 yaptık. Amacı, tagları atarken kolaylık sağlayacak ufak bir trick üretmek.
* Ortadaki kutulardan bir prefab oluşturduktan sonra bunu sunucuda tek tek oluşturmana gerek yok. Resources’te bulunması yeterli.

## 19-Kutuma Dokunma-XVIII

* Can barını ve darbe almasını sunucu tarafa taşıdık. Bunu GameKontrol’u servere uyarlayarak yaptık, içindekileri değiştirip Resources’e attık. Şimdi birbirlerine verdikleri zararlar ikisinde de görünüyor.

## 20-Kaynak Dosya-II

## 21-Kutuma Dokunma-XIX

* \*G :: 3 adet tahtayı birleştirdi ve yalnızca parentlarına gerekli componentleri ekledi, yani ‘DuzTahta’ya. Bu yeterli gelir.
* Animasyon altına Animator Controller ekledi ve bunu ‘DuzTahta’nın Animator componentine verdi.
* \*B :: Resources’te oluşturulacak ve yok edilecek objeler olmalı.

## 22-Kutuma Dokunma-XX

* Vurulduğun can dolduran küpleri yaptık.

## 23-Kutuma Dokunma-XXI

* \*B :: GameKontrol scripti her iki oyuncuda da çalışacağından, tek çalışması gereken metotlara if(PhotonNetwork.IsMasterClient) ekledik.

## 24-Kutuma Dokunma-XXII

* ‘Odul’ canlarının oluşması ve yok olmasını yaptık.
* Şu an giriş ekranında lobiye bağlanmadan odaya girişe basabiliyoruz. Bu hataya sebep verebilir, düzeltilmeli.

## 25-Kutuma Dokunma-XXIII

* İstatistik ve puan sistemini yapacağız.

## 26-Kutuma Dokunma-XXIV

* Serverla alakalı butonları geliştirdik, bağlantı olmadan tıklanamaz yaptık.

## 27-Kaynak Dosya-III

## 28-Kutuma Dokunma-XXV

* İstatistikleri işler hale getirdik ve oyun sonu panelini tasarladık.

## 29-Kutuma Dokunma-XXVI

* OyunSonuPanel’in çıkmasında sıkıntı oldu ve bunun server taraftaki bir gecikmeden kaynaklandığını düşündü. Bu sebeple GameKontrol script’inde foreach ile deaktif nesneleri de yakalayabilen bir algoritma kurdu ve bunu oradan çözdü.
* Oyuncu.cs, locak bir script olduğundan, kazanma ve kaybetme durumunda her iki kullanıcı da ayrı ayrı çalışır ve bu tutarsızlıklara sebep olur. Bunun çözümü veritabanı kullanmak aslında. Ancak bunsuz da, zor olsa da, bir çözüm üretilebilir. Bunu ‘sonuc’ metodunda kurduk.

## 30-Kutuma Dokunma-XXVII

* OyunSonu ekranına son halini verdik. Daha sonra Androide geçtik ve uyumu için birkaç düzenleme yaptık.

## 31-Kutuma Dokunma-XXVIII

* Topun topa çarpmasını ve oyuncunun kendi kendine zarar verememesini sağladı.
* Çıkışın butonla mı olduğunu sorgulayarak ilgili işlemleri revize etti. Terk durumunda galibiyet ayarlamasını böylece halletti.

## 32-Kaynak Dosya-IV

# Bölüm-26: Örnek Oyun-3D-PC-Third Person-Suikast

## 1-Suikast 1

* Terrain oluşturdu, yükseltiler oluşturdu ve üstlerini kapladı. Yolları döşedi, ağaçlandırma yaptı.

## 2-Suikast 2

* Çitleri ve kayaları yerleştirdi.

## 3-Suikast 3

* Evlerin içini “Point Light” ile aydınlattı.

## 4-Suikast 4

* Karakter yönetiminde “LateUpdate” kullanılır. Bu, daha yumuşak geçişler sağlar.
* Miximo’dan indirdiğin kaynaklarda animasyonları skinsiz indirirsen animasyon sırasında texture’da yırtılmalar olabilir.

## 5-Suikast 5

* Animatörde ağaç yapısı oluşturdu. Böylece şartlı dallanma ile yürüme ve koşma animasyonları arası geçiş yapabilecek.
* Standart Asset’teki FreeLookCameraRig’i kamera olarak kullandı. Bunun kullanımı ve script tarafı farklı ve detaylı işlemler içeriyor.

## 6-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 1.FULL KAYNAK DOSYASI

## 7-Suikast 6

* Silahın karakterin Mesh’inin child’i yaparsan karakterin animasyonuyla uyumlu hareket etmez. Bu sebeple karakterin kemik sistemi altına almalısın.

## 8-Suikast 7

* Silahın ses, efekt ve ateş etme sıklığını ayarladık.

## 9-Suikast 8

* Silahın mermi sayısını ayarladı ancak şarjör değiştirme işlemlerine başlamadı.

## 10-Suikast 9

* Yine reload algoritmasını implemente etti.

## 11-Suikast 10

* Şarjör algoritmasını tamamladı.
* Aldığı animasyonlar “ReadOnly” olduğundan event ile tetikleyemedi. Bunun yerine animasyonun script’ine bir bool tetik koydu ve silah script’inde bunu kontrol ettirerek işlem yaptı.

## 12-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 2.FULL KAYNAK DOSYASI

## 13-Suikast 11

* Animatörde ağaç yapısı yerine parametreli çalışmak daha profesyonel.
* Eğer paralel olarak çalışması gereken animasyonlar varsa, Animatöre layer ekleyebilirsin. Örneğin reload yaparken sola yürümek gibi. Aynı layerda olurlarsa biri çalışırken diğerine geçemez.
* \*G:: Bunu yaptığında reload çalışmaz. Çalışması için katmanın “Ağırlığını” arttırmalısın. Ayrıca “Blending”i değiştirdiğinde de animasyonların harmanlanma şekli değişir.

## 14-Suikast 12

* Animatörde float değerlere göre değişen animasyonlar ayarladı ve karakterin sola meyilli, geri sola meyilli vb. yapabilmesini sağladı.

## 15-Suikast 13

* Hareketten başka bir harekete geçişte “Idle”ın araya girmemesi için hareketler arasında birçok bağlantı oluşturdu. Bu baya akıcılık sağlar.

## 16-Suikast 14

* Namespace oluşturdu. Bununla diğer classlardaki kod kalabalığını önledi.

## 17-Suikast 15

* Namespace’i geliştirmeye devam etti.

## 18-Suikast 16

* Namespace’te kodları toparlamaya devam etti.

## 19-Suikast 17

* Namespace’te kodları toparlamaya devam etti.

## 20-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 3.FULL KAYNAK DOSYASI

## 21-Suikast 18

* Düşmanların dış görünüşlerini ayarladı ve animasyonları çıkarttı. Navigasyonu ayarladı. Script’i yazmaya geçmedi.

## 22-Suikast 19

* Ajanın belli bir mesafeden oyuncunun menzile girdiğini anlaması ve ona doğru hareket etmesini ayarladı. Bunun için birkaç fonksiyon kullandı. Temel fonksiyon olarak Overlap kullandı. Ayrıca ajanın tespit mesafesini bir Gizmos fonksiyonuyla oyunu başlatmadan çizdi.

## 23-Suikast 20

* Ajanın menzilinden çıkıldığında ajanın tekrar başlangıç noktasına dönmesini sağladı. Ayrıca Quaternion.Euler() ile rotasyonunu da ayarladı.

## 24-Suikast 21

* Ajanın menzilini 2 kademe yaptı. Birinci kademede takip, ikinci kademede çatışma başlıyor.

## 25-Suikast 22

* Navmesh.isStopped=true diyerek ajanı durdurdu, ajan nişandayken kayma sorunu giderildi.
* Debug.DrawRay() ile Raycast ışınını görebilirsin.

## 26-Suikast 23

* 3 adet devriye rotası oluşturdu. Ajanın bunları sırayla dolaşması için bir algoritma tasarladı.

## 27-Suikast 24

* Noktaları “!=” ile sorgulamak işe yaramaz, ajanın nokta etrafında dönmesine sebep olur. O yüzden “Vector3.Distance()” kullandı.

## 28-Suikast 25

* 3 devriye rotası oluşturmuştu. Bunların random seçilmesini sağladı.
* Devriye turunun bitiminde belli bir süre beklemek için bir Coroutine daha ekledi.

## 29-Suikast 26

* Devriye sisteminin geliştirilmesine devam etti.

## 30-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 4.FULL KAYNAK DOSYASI

## 31-Suikast 27

* Ajanın devriyesini tamamladık. Şimdi ajanın silah ve hasar verme işlemlerine başladı.

## 32-Suikast 28

* Ajanın menziline girdiğimizde bize dönmesini sağladı.
* Ajanın 2 adet menzili mevcut Şüphelenme ve ateş etme.
* Ajana silah namlusunda bir Raycast ışını çıkış noktası belirledi.
* Ajanın bize dönmesinde yönü ayarlamak için: (Kaynak Vektör – Hedef Vektör) doğrultusunu kullandı.

## 33-Suikast 29

* Ajanın ateş etme postürünü daha önceki videolarda ayarlamıştı. Şimdi de silahından çıkan merminin yönünü ve sıklığını ayarladı. Ayrıca mermi çarptıkça karakterin can değerinin azalmasını sağladı.

## 34-Suikast 30

* Düşmana artık hasar verebiliyoruz ve canı 0 olduğunda düşman ölüyor.

## 35-Suikast 31

* Ajanın silahının ses ve efektini ayarladı.
* Karakterimiz için health bar yaptı.
* Bir tane daha ajanı ve patronu ekledi.

## 36-Suikast 32

* Ajan2’nin silah tutuşunu ayarladı.
* Ajan2’nin ateş etmesinde sıkıntı çıkıyordu. Silah rotasyonunu bu ajana uygun ayarlayınca karakteri vurabildi.
* Ajanların script’inde bir çakışma mevcut. Bu sebeple ajanların ateş etmesinde sıkıntı çıkıyor.

## 37-Suikast 33

* Bir ajanın silah rotasyonunu ayarlayınca ateş edebildi ancak tüm ajanlar için sorun çözülmüş değil.
* Devriye tetikleriyle ilgili bir sorun çözüldüğünde sorun çözüldü.
* Düşmana ateş edildiğinde düşmanın karaktere doğru koşmasını sağladı. Bunun için şüphe menziline girince koşmayı bırakıp şüphe ve devamının çalışacağı bir algoritma tasarladı.

## 38-Suikast 34

* Win-Lose panelinde cursoru kitlemek için tek satırlık bir kod kullandı.

## 39-Suikast 35

* Ana menüyü ve kazanma-kaybetme ekranlarını tasarladı, butonları ayarladı.

## 40-Suikast 36

* Patronun odasına girildiğinde artık kazanma ekranı aktif oluyor.
* Ajanları arttırdı, menzillerini ve hasarlarını değişken yaptı.

## 41-Suikast 37

* Ajanlar ateş ettiğinde hata aldı. Bunun Raycast’ın çarptığının Player olup olmadığını “if” ile kontrol etmesiyle düzeltti.

## 42-Suikast 38

* Tüm kullanılan materyallerde “Enable GPU Instancing” i aktifleştirdi. Bu performansı arttırır.
* Ayrıca sahnede kullanılmayan dosyaları kaldırdı.

## 43-OYUNUN BU AŞAMAYA KADAR OLUŞMUŞ OLAN 5.FULL KAYNAK DOSYASI

# Bölüm-27: Örnek Oyun-2D-Mobil-Hafıza Oyunu-Oyun Adı:Ben Bulurum

## 1-Ben Bulurum-I

* 2D Sprite Pack import etti.
* Image’ye Button eklersen tıklanabilir hale gelir.

## 2-Ben Bulurum-II

* Temelleriyle uğraştık.

## 3-Ben Bulurum-III

* \*G :: Eğer butonla obje ve değer göndereceksen önce objeleri gönder. Aksi halde hata alırsın.

## 4-Ben Bulurum-IV

* Kutulara bastığımızda 1 saniyelik bir bekleme ayarladık.
* Açılan yerin tekrar tıklanamaması ve 2’den fazla kutunun aynı anda açılamamasını yaptık.
* Image dizisinde dolaşan foreach, silinen objeler null olacağından hata verir. Bunu basitçe if(item!=null) ile çözdük.

## 5-Kaynak Dosya-I

## 6-Ben Bulurum-V

* ‘Grid Layout Group’u ayarladı. Böylece farklı boyutlara uyum sağlandı.
* Grid’de Destroy veya deaktif etme işlemlerinde kayma olur. Bu sebeple gereken componentleri pasife çektik.

## 7-Ben Bulurum-VI

* Sayacı yaptık ve ‘Win’, ‘Game Over’ ekranlarını tasarladık.
* \*A :: Ana menü ve Tekrar Oyna’yı tasarladık. Bir sahneye geçiş yapabilmek için sahne ‘Build Settings’da ekli olmalı.

## 8-Ben Bulurum-VII

* Mobile Game Icon Pack’ı import etti.
* Ana menüden geçişte müziğin yok olmaması için ‘DontDestroyOnLoad’ kullandık. Geri ana menüye geçtiğimizde ses objesi çiftlendi ve güzel olmadı.

## 9-Ben Bulurum-VIII

* Bunu çözdük ancak ‘Hemen Başla’ çalışmadı. Bu yüzden ilgili yerleri kestik. OyunKontrol’den ve AnaMenuSes’e attık. Bunun sebebi, butonun script’inde ekleme-çıkarma-yok etme olmasıdır. (Soru-8)
* Project Settings/Player’da son dokunuşları yaptık.
* Scripting Backend :: IL2CPP ve ARMV7-ARM64 tikli olmalı Google Play için.

## 10-Ben Bulurum-IX

* Durdurma ve devam etme butonlarını yaptık. Çıkış panelini dizayn ettik.
* AnaMenuKontrol’deki Start içindeki Time.TimeScale=1 olmazsa, diğer scriptlerdeki zamanı durdurmalar oyunu patlatıyor. Sayaç çalışmıyor, tıklanmıyor.

## 11-Kaynak Dosya-II

## 12-Ben Bulurum-X

* Zamanlayıcıyı slider ile yeniden tasarladık.

## 13-Ben Bulurum-XI

* Prefab bir objeye normal sahne objelerini eşleyemezsin. Componentlerin elemanları da prefab olmalı Burada prefab çok yük bindiriyor. Bir sürü oluşturma ve işlem gerekiyor meteorları(gizleyicileri) halletmek için.
* Randomize gelmelerini hallettik.

## 14-Kaynak Dosya-III

# Bölüm-28: Örnek Oyun-3D-PC-Araba Yarışı-Speedy Gonzales

## 1-

## 2-

## 3-

## 4-

## 5-

## 6-

## 7-

## 8-

## 9-

## 10-

## 11-

## 12-

## 13-

## 14-

## 15-

# Bölüm-29: Bonus

## 1-

## 2-

## 3-

## 4-

## 5-

## 6-

## 7-

## 8-

## 9-

## 10-

## 11-

## 12-

## 13-

## 14-

## 15-

# Alfa-Beta-Gama

## 1-Alfa

* 2.5 D, 2 boyut kamera açısına sahip, 3 boyutlu tasarlanmış oyun.
* Nesneyi pasif yaparsan ona atanmış script çalışmaz.
* Space.Self’te hareket için kendini referans alır. Space.World’de çevreye göredir. Gerçek dünya gibi olanı world’dür.
* Texture’da metalik yoksa, metalik yerine ‘Roughness’i koyabilirsin.
* Textures’in Inspector’unda aşağıdaki ‘Use Crunch Compression’, objeyi sıkıştırırken işlemciyi de devreye sokar.
* IDragHandler ve IEndDragHandler’i gördük. Bunlar Event Trigger ile ilişkili ve sürükleme olaylarında kullanılırlar. Bununla basit bir item çantası yaptık.
* Raycastı silahtan gönderirsen ışın yanlış nesneleri vurabilir. Bu sebeple ışını kameradan göndermelisin.
* PlayerPrefs kullanmadan önce Edit altından eskilerini temizleyebilirsin.
* Ne kadar kaliteli texture kullanırsan kullan, buradaki ayarları doğru yapamazsan istediğin sonucu alamazsın.
* Eğer çarpışma yakalamayacaksan box collider’ı kaldır.
* Resources’dekiyle aynı isimde PreFab bulunmasını Photon istemiyor.
* Ana menü ve Tekrar Oyna’yı tasarladık. Bir sahneye geçiş yapabilmek için sahne ‘Build Settings’da ekli olmalı.

## 2-Beta

* #region kullan. Kodun çok daha derli toplu gözükür.
* Objeyi pasif yapmak istiyorsan üstteki checkbox’taki tiki kaldır. Oyun çalışırken pasif yaparsan OnDisable çalışır.
* Coroutine’u dizayn ettikten sonra private değişkene atayarak kullan. (Coroutine’yi Privateye Ata)
* Component yakalamayı değişkene ata. Aksi halde performans kaybedersin.
* ‘FindWithTag’ı gördük. Daha iyi bir seçenektir. Aynı tagda birden çok taglı obje varsa bunlardan ilkini yakalar.
* Bir obje pasifse hiçbir işlemini gerçekleştiremez. Buna kendini aktifleştirmek de dahildir.
* <> = (typeof())
* Rotasyon mevcutsa, düz ilerletmek istesen de, doğal olarak düz gitmez.
* Round metodu ‘.5f’ de sayı tekse veya çiftse farklı şekilde davranır.
* .tag==”” yerine CompareTag() kullan.
* Kinematic obje, kinematic veya statik objeler ile çarpışamaz. Sadece dinamik objelerle çarpışır. ‘Use Full Kinematic’ seçeneği ile bu durumu tüm çarpışmaları yakalayacak şekilde değiştirebilirsin.
* Dinamic objeler tüm türlerdeki objelerle etkileşime girebilir.
* Edge Collider içi boş, Polygon Collider içi dolu.
* Composite Collider diğer colliderları birleştirmemizi sağlar. Herhangi bir colliderı dahil etmek için ‘Used by Composite’yi seç
* Read/Write Enabled’i işaretleyerek Texture özelliklerine scriptten erişebilirsin. İhtiyaç yoksa açmamalısın.
* sRGB’yi çok iyi detaylanmış bir objen varsa kapat.
* Texture’deki ‘Aniso Level’, dokuya dik olarak yaklaştıkça kalitesinin artmasını sağlar. Arttırdıkça performans düşer.
* Duvar gibi objelerin arkasındaki objelerin, açıya girmeden GPU’ya gitmesi gereksizdir. Bu sebeple Inspector’daki obje adı yanındaki ‘Static’ kısmını ‘Occuluder Static’ seçmelisin.
* Tüm materyalleri başka bir pipeline’a uygun geliştirmek-değiştirmek için, Edit/Render Pipeline/… Bazı nesneleri yine elle değiştirmen gerekir.
* En alt bloğa ‘Orient:Face Camera Position’ eklersen efektin 3 boyutluymuş gibi görünür.
* Klipler arasında değişiklik yapacaksan, herhangi birinde kullandığın componenti silme. Componenti istemediğin yerde deaktive edebilirsin.
* Canvası 2D’den çıkarıp 3D öğelere koyabilmek için ‘Render Mode’u ‘World Space’ olarak seçmelisin.
* Agentların trigger’ını açsan dahi trigger’ı açık yerlerde düşmezler.
* Inputları FixedUpdate ile yakala. Bu Update’den daha hızlıdır.
* Terrain Inspector’deki ‘Draw Instanced’ Scene’deyken hesaplanmayıp Game’deyken hesaplanan detayların Scene’deyken de hesaplanmasını sağlar.
* En az performans gerektiren ‘Anti-Aliasing(Kenar Yumuşatma)’ yöntemi FXAA’dır. Bu yöntem ışık yoğunluğu fazla olan yerlerde iyi çalışmaz. SMAA ise daha kaliteli sonuçlar verebilir. Ancak %90 FXAA kullanılır. TAA ise hareketli objelerin kenarlarını yumuşatmak için kullanılır.
* LayerMask ile ışının hangi hedeflerle etkileşime girebileceğini bulabilirsin. Scriptteki 1<<8 ifadesi sadece 8. katmanı dikkate al demek. ‘x=~x’ ile kuralı tersine çevirebilirsin.
* Resources’te oluşturulacak ve yok edilecek objeler olmalı.
* GameKontrol scripti her iki oyuncuda da çalışacağından, tek çalışması gereken metotlara if(PhotonNetwork.IsMasterClient) ekledik.

## 3-Gama

* Yeni öğrenenlerin yaptıkları en büyük hata, acele etmektir. Oyun geliştirmeyi kolay olarak düşünüp hızlı para kazanmak istersen kaybedersin. Asla acele etme.
* ArrayList’in kullanılması artık önerilmiyor. List<> kullan.
* GetComponent, eğer aynı türde birden fazla component varsa ilkini yakalar.
* ‘Equals’ metodu her zaman doğru çalışmaz vektörlerde. Vektörlerin ‘==’ ile karşılaştırılması gerekir.
* transform.LookAt’ i gördük. Bu sanırım kameranın sürekli bakmasını sağlıyordu.
* transform.Find ile nesneyi yakaladın ve adını değiştirdin. Sonra değiştirdiğin çocuğuna ilk adı kullanarak ulaşamazsın. Değistirdiğin adla ulaşabilirsin.
* Fırlatılacak objeyi ‘prefab’ klasöründen atadım. Direkt var olan nesneyi oraya atarsan, nesne destroy edildikten sonra exception yersin.
* IsTrigger, Box Collider’da açıkken, nesne diğer nesnelerle çarpışmaz. Ama çarpışmaları yakalar.
* Şu an birden fazla agent türü tanımlayabilsen de sadece biri çalışıyor ve bu Unity’nin bir hatası. Düzeltilmiş olabilir.
* Mermi yaparken prefab’tan oluşturup Destroy ile süreli yok etsen de performanstan çok kaybedersin. Bu sebeple Raycast kullanabilirsin. Vurduğun nesnenin Rigidbody’sine erişip ona güç uygulamak iyi bir yol.
* Cloth Componenti’i ekledi. Eklenen objeden meshi kaldırmalısın.
* İşin içine sonrada server gireceğinden, her şeyi GameKontrol’de toplamaya çalışıyor. Yoksa oyuncu sağlığı, oyuncu nesnesinde de tutulabilir.
* Dumanpuf efektinin scriptine gidip ‘Destroy’ metodunu ‘PhotonNetwork Destroy’ olarak değiştirdik. En sonda Photon’daki garip bir hata ile karşılaştık ve bunu ‘PhotonNetwork.Destroy’u if(pw.IsMine) içine alarak çözdük.
* 3 adet tahtayı birleştirdi ve yalnızca parentlarına gerekli componentleri ekledi, yani ‘DuzTahta’ya. Bu yeterli gelir.
* Eğer butonla obje ve değer göndereceksen önce objeleri gönder. Aksi halde hata alırsın.

Tips!

1- 100 Unity Tips

<https://www.youtube.com/watch?v=thA3zv0IoUM&list=WL&index=1&t=194s>

Araştır! Pixel perfect kamera araştır.

Araştır! Struct-Class farkını araştır.

Araştır! [FormerlySerializedAs(“string”)]

Araştır! InvokeRepeating

Araştır! StringBuilder

Dene! Script+Animator ile nesnenin boyutu küçülürken rengini değiştirdi. Video-12.tip.

Dene! Animasyonda birçok frame seçiliyken ‘F’ ye bas. ‘C’ ye basınca Curve View’a geçiyor. ‘K’ ye basınca KeyFrame’e geçiyor.

Dene! Scene sekmesinin altındaki butonların en sağındaki, sahnedeki efektleri elimine ederek sahneyi görmeni sağlar.

Dene! [System.Serializable]

Dene! Color picker diye bir şey var, dışarıdan renk alabilmeni sağlıyor.

Dene! qwe rnt ile obje ayarlarını seçebiliyorsun mouse ile tıklamak yerine.

Dene! Rect Transform Tool ile pivottan büyütmek yerine sadece çektiğin yerden büyütürsün.

Dene! Ctrl basarken obje sürükle. Buna snapping denir. Ayarlarına Edit/Snap Settings ten ulaşabilirsin.

Dene! V basarken obje sürükle.

Dene! Edit/Selection ile o an seçili nesnelerin seçililiğini kaydedebilirsin. Sonra yeniden bunları seçmek yerine oradan direkt tıklayabilirsin.

Dene! Oyunu || ile durdurduktan sonra hemen yanındaki butona bastığında 1 frame ilerletirsin.

“Project Settings/Graphics/Transparency Sort Mode” ile layer önceliğini değiştirebilirsin. Varsayılan olarak Z ekseninde önde olan daha önde gözükür, sen bunu örneğin Y eksenine verebilirsin.

Destroy(x,y)’de y olarak zaman verebilirsin. Böylece süreli bir yok etme elde edersin.

\*G:: Animasyon speed’ini -1 yaparsan terse akar.

Vector3.Distance kullanma. Onun yerine manuel olarak karekök al. Bu daha hızlıdır.

\*B:: private değişkenleri Inspector’da göstermek için değişken üstüne [SerializeField] hide flag’ini ekleyebilirsin.

\*B:: public değişkenleri Inspector’da gizlemek için değişken üstüne [HideInInspector] hide flag’ini ekleyebilirsin.

\*B:: Unity’ye klasör kısa yolu ekleyebiliyorsun. Bu işini baya kolaylaştırabilir.

\*A:: Obje seçiliyken ‘F’ basarsan objeye focuslanır. Eğer 2 kere ‘F’ basarsan hem focuslanır hem de obje hareketini takip eder.

\*G:: Kamera seçiliyken Ctrl+Shift+F basarsan kamerayı o anki bakışa ayarlar. Çok kullanışlı.

\*B:: Project sekmesinin search barında t:scene, t:texture, t:script gibi aramalarla yalnız o kategoriye ait şeyleri çekebilirsin.

\*B:: Script’inde bir satırı aşağı veya yukarı taşıman gerekiyorsa, Alt+(Up-Down) kullanabilirsin. Kopyala-yapıştıra göre çok daha etkindir.

\*A:: Unity’nin o component ile alakalı dokümantasyonuna, component yanındaki kitaba basarak ulaşabilirsin.

\*B:: Child’ları olan bir objenin tüm soyunu açmak için Alt’a basılı tutarak tıkla. Collapse için de açılıyken aynısını uygulayarak halledebilirsin.

\*G:: Ekranları keyfine göre düzenledikten sonra bu layout’u kaydedebilirsin. Üstteki ‘Layout’ sekmesine tıkla.

\*B:: [MenuItem(“…”)] ile üst menüye elemanlar ekleyebilirsin.

\*A:: Layer eklerken Effect/Light tarzı bir ekleme yaparsan Efect altına Light layeri eklemiş olursun.

\*B:: Ekranı büyütmek için Shift+Space bas.

\*B:: Inspector’u lockladıktan sonra ana ızgaraya çekerek çiftleyebilirsin. Böylece 2 tane Inspector aynı anda açık kalır, istediğin değerleri birinden diğerine çekebilirsin.

\*A:: Inspector’un hemen altındaki simgeye tıklayarak objeye bir gizmos atayabilirsin.

\*G:: WaitForSeconds eğer TimeScale 0 yapılırsa yine de durmaz. Böyle bir durumda durması için WaitForSecondsRealTime kullanmalısın.

\*B:: Game’deki Mute ile sesleri susturabilirsin.

