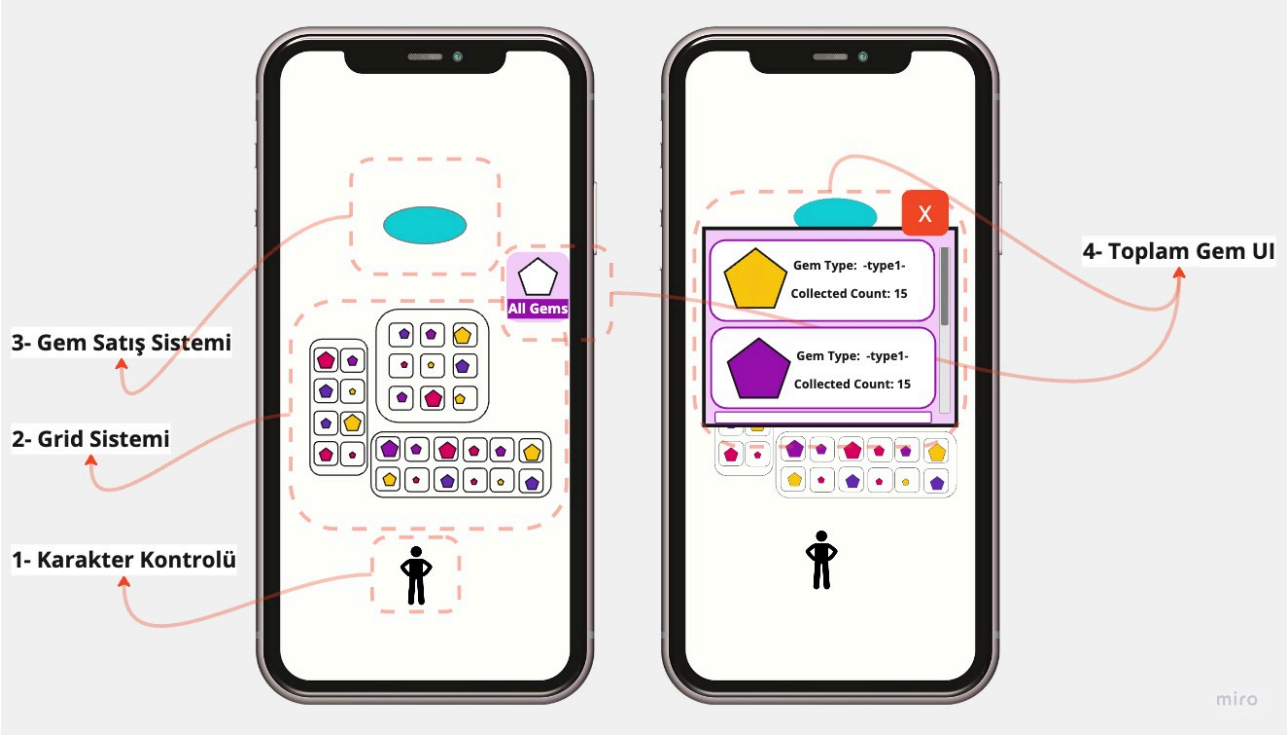
FLUSH GAMES - GAME DEVELOPER CASE

Oyunumuz, tarladan otomatik yetiştirilen farklı tip gem'lerin toplanıp satışının yapılmasıdır.

Rastgele gemler, gridler üstünde oluşur. Joystick ile hareket eden karakter, gemlerin üzerine gelerek onları sırtında stack olarak tutar. Karakter, satış alanına girince sırtında stacklediği gemleri yok ederek satmış olur.

Ekran üzerinde bulunan bir buton ile oyuncu yaptığı bütün satışların sayısını bir UI ile görebilir.



1- KARAKTER KONTROLÜ

A- Hareket

Stickman karakterimiz, idle ve walk animasyonları ile verilecektir. Karakter mobil builde uyumlu bir joystick ile hareket etmeli. Altta linki verilen free asset kullanılabilir.

https://assetstore.unity.com/packages/tools/input-management/joystick-pack-107631

B- Stack

Karakter, yerden topladığı gemleri sırtında stacklemeli. Objelerin tam sırtına denk gelmesi gerekmiyor. Arkasında havada durabilirler. Sınırsız stack yapabilir.

2- GRID SİSTEMİ

A- Grid Yapısı

Gemlerin yetişeceği noktaları belirtilen gridler gerekiyor. Gridlerin, satır ve sütun sayısı NxN olarak, **kodda değişiklik yapmadan, Inspector üzerinde** girilebilir olmalıdır.

Eğer birden fazla grid bloku üretilmek istenirse, scriptin üzerinde olduğu boş bir obje kullanmak yeterli olmalı. Örneğin 4x3'lük bir grid üretmek istersek, boş bir objeye scripti atıp değerleri girerek üretebilmeliyiz.

Gemlerin yetişeceği Tile zemini için düz plane kullanılabilir.

EKSTRA: Gridler Runtime yerine Editor üzerinde generate edecek bir editor kodu olursa artı olarak değerlendirilecek.

B- Gem Yetiştirme

a- Gem çeşidi

Grid içindeki her tile üzerinde gem büyümeli. Bu gemlerin çeşidi rastgele bir listeden seçilmelidir.

Birden fazla gem çeşidimiz olacak. Bir gem çeşidi şu bilgileri içermeli:

-Gem Adı
-Gem Başlangıç Satış Fiyatı
-Gem ikonu
-Spawnlanacak Gem modeli

Bu bilgiler Inspector üzerinde eklenebilir olmalı. Inspector üzerinde, sonradan istersek yeni gem tipi ekleyebilmeliyiz.

Örnek üç gem FBX olarak verilecektir. Ayrıca UI'da kullanabilmek için üç tane 2d icon verilecektir.

Test etmek için kopyalayıp renklerini değiştirip yeni gem çeşitleri oluşturabilirsiniz.

b- Spawn yapısı

Her tile üzerinde rastgele bir gem spawn olur. Gemlerin başlangıç scale sıfırdır. 5 saniye içinde gem boyutu 1 birime çıkar.

Bir gem 0.25 birim scale olduğunda toplanabilir olur. Bir gem toplandıktan sonra yerine yine rastgele 0 scale bir gem gelir. Toplanan gem sırta stacklenir.

3- GEM SATIŞ SİSTEMİ

Karakter, sırta stacklediği gemlerle satış alanına gider. Satış alanına girince sırtındaki stack'in en üstünden itibaren satış yapılır. Satış alanını materyali %50 transparan olan büyük bir küp veya silindir yapabilirsiniz.

Karakter alan içinde durdukça 0.1 saniye aralıklarla en üstten itibaren gemler yok olur. Eğer karakter alandan çıkarsa yok olma işlemi durur.

Yok olan gemin üzerinde tutulan (Başlangıç Satış Fiyatı + scale Birimi * 100) kadar gold kazanılır. Örneğin başlangıç fiyatı 100, scale değeri 0.37 olan bir gem 137 gold kazandırır.

Toplam Gold değerini sol üst UI kısmında yazdırabilirsiniz.

4- TOPLAM GEM UI

Ekranın orta sağ köşesinde bir buton olmalı. Butona bastığımızda UI içeriği bir pop-up olarak açılmalı.

Açılan pop-up üstünde kapatma butonuyla kapatılabilmeli.

A- Pop-up İçeriği

Pop-up içinde, tüm gem çeşitlerinden kaç tane topladığımız bilgisi olmalı. Yukarıda verdiğimiz gem çeşitleri bilgileri burada olmalı. En üstteki şemadaki gibi her gem çeşidi bir satır gibi olmalı. Bu satır içinde icon, gem ismi, bu gemden toplanan sayı bilgileri olmalıdır.

Pop-up içeriği scroll edilebilir olmalıdır. Dinamik olarak yeni gem çeşitleri ekleyebildiğimiz için her gem çeşidi alt alta eklenmeli. Scroll ile de bu gemlerin satırlarını görebilmeliyiz.

5- KAYIT SİSTEMİ

Pop-up içine yazdırdığımız toplanan gem sayıları ve toplanan gold sayısı kaydedilmelidir. Oyun kapatılıp açılsa bile bu değerler kaldığı yerden başlamalı.

ÖNEMLİ NOTLAR

Oyunun son hali **Unity** proje dosyası olarak Github'ta public repo olarak paylaşılmalı. Unity 2021.3.15f1 sürümünün kullanılmasını tavsiye ederiz.

Oyunun test edilmesini engelleyen major bir hata olmaması gerekir. Test edilemeyen case maalesef değerlendirilmeyecektir.

OOP ve Design Pattern'lere dikkat edilmesini öneririz. Singleton, serializable class gibi yapılar artı olarak değerlendirilecektir.

Oyunda görsel tarafa dikkat edilmeyecek ama genel anlamda hypercasual - hybridcasual oyunların görsel diline yakın sahne düzeni, ışık, renk, font kullanımı artı olarak değerlendirilecek.

Obje hareketlerinin smooth olmasını tavsiye ederiz. Hypercasual - hybridcasual oyunlarda kullanılan hareketlere benzer etkiyi görmek iyi olur. Bu hareketler için DOTween asseti kullanılabilir.

https://assetstore.unity.com/packages/tools/animation/dotween-hotween-v2-27676

Önerdiğimiz assetler dışında mümkün olduğunca az asset, plug-in kullanmayı tavsiye ederiz. Case ile amacımız anlaşılabilir temiz kod yazma bilgisini ölçmektir. Genel olarak oyuna kod tarafını bilmeyen bir level designer'ın rahatlıkla level yapabileceği bir yapı bekliyoruz.

Unity içerisinde klasörlemeye dikkat edilmesini öneririz. Örneğin klasörleme şöyle olabilir:

Dev
-Scenes
-Scripts
-Prefabs
Art
-Models
-Sprites
-Materials

Case ile ilgili sorularınız olursa sorabilirsiniz. Başarılar!