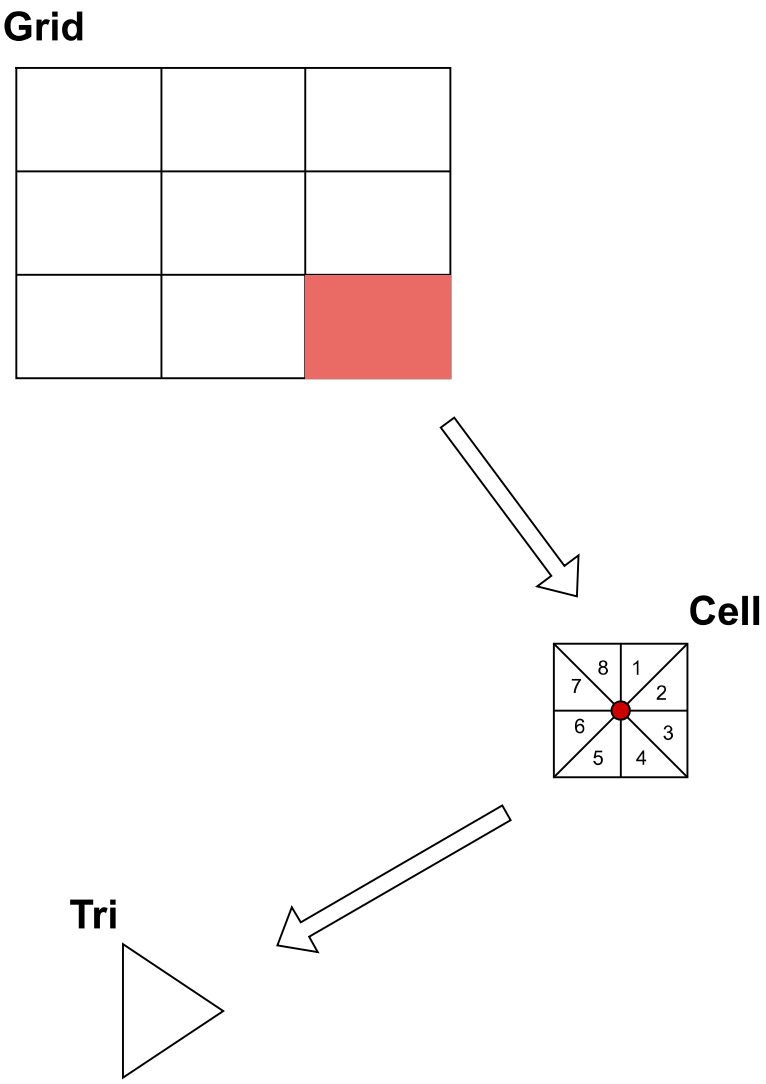
Algoritma (High Concept)

Puzzle alanını kontrol etmek amacı ile iki boyutlu ızgara (grid) yapısı kullanılmıştır. Grid alanı case'de belirtildiği gibi [4,6],[4,6] boyutlarında belirlenebilir.

Grid standart yapıya uygun olarak hücrelerden (cell) oluşur. Buna ek olarak her bir cell, puzzle parçalarındaki çeşitliliği arttırmak amacı ile sekir parçaya (tri) bölünmüştür.

Grid, Cell ve Tri



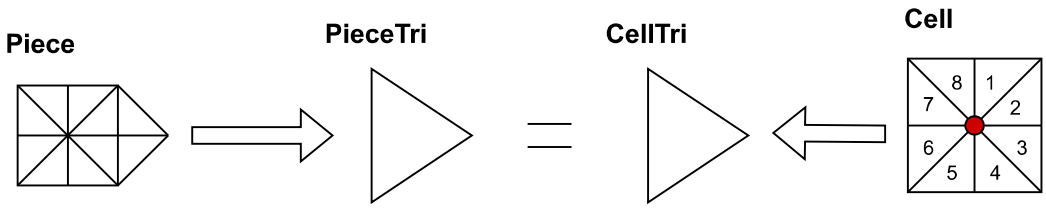
Her tri 3 köşe nokta ve bir ağırlık merkezinden oluşur.

```
Tri
 {
      Vector3[] vertices;
      Vertor3 [] origin;
}
```

Her tri'nin ilk köşe noktası bağlı olduğu hücrenin merkezine eşittir. Tri'nin bu özelliği bağlı olduğu hücreyi aramakta kolaylık yaratmaktadır.

Piece ve Tri

Grid üzerine yerleştirmek üzere hazırlanmış puzzle'ın parçaları proje boyunca piece olarak isimlendirilmiştir. Gridin en küçük parçası olan tri'lerden oluşur ve level data'sından yüklendikten sonra sahnede procedural olarak oluşurlar.



Piece'lerin sahip oldukları tri'lerin boyutları grid'de yer alan trilerin boyutları ile bire bir aynıdır. Bu özellikleri grid'e yerleştirme algoritmasında kullanılacaktır.

Piece Data

Seviye yüklenirken oluşacak olan parçanın mesh bilgileri PieceData isimli modelde saklanır. Piece MonoBehaviour'ı bu modeli okuyarak parçanın meshini procedural olarak oluşturur.

Bu modele daha yakından bakacak olursak:

```
PieceData
{
	Vector3[] vertices;
	int[] triangles;
	Vector3 origin;
}
```

vertices: LevelCreator tarafından derlenmiş vertex dizisi. Mesh'e doğrudan aktarılabilir.
triangles: LevelCreator tarafından derlenmiş triangles dizisi. Mesh'e doğrudan aktarılabilir.
origin: Mesh'in pivotudur. Parça grid'deki konumunda değil local zero'da oluşturulmak isteniyorsa vertex'lerden bu değer çıkartılmalıdır.

Piece'in Grid'e Yerleşmesi

Bir piece grid üzerinde bırakıldığında sahip olduğu bütün tri'ler için grid üzerindeki en yakın tri tespit edilir. Bulunan CellTri'nin hali hazırda başka bir parça ile doldurulmuş olması durumunda işlem başarısız kabul edilir ve parça başlangıç noktasına geri döner.

Parçanın grid'e yerleştirilebilmesi için parçanın sahip olduğu her tri'ye karşılık grid üzerinde boş ve birbirinden farklı bir tri var ise işlem başarılı kabul edilir ve grid'e yerleşir.

Level data'da yer alan piece sayısı ile grid'e yerleştirilmiş grid sayısı birbirine eşit olduğunda seviye tamamlanır.

Level Data

Oyun için oluşturulmuş her bir seviyenin (level) bilgileri LevelData isimli modelde saklanır.

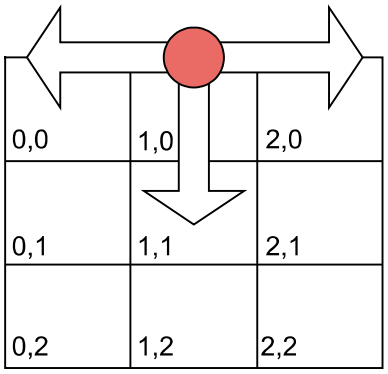
Bu modele daha yakından bakacak olursak:

```
LevelData
{
    Vector2Int gridSize;
    LevelDifficulty levelDifficulty;
    PieceData[] pieces;
    Vector3 origin;
}
```

gridSize: Seviyenin grid size'ı [4,6],[4,6] boyutlarında olabilir.
levelDifficulty: LevelCreator'da seviye üretilirken parça sayısına göre dinamik olarak oluşturulur.

- [5,6]: easy
- [7,9]: medium
- [10,12]: hard

pieces: Seviyede oluşturulacak olan parçaların dataları.
origin: Grid'in konumlanacağı merkez. Hücreler merkez üzerinde aşağıdaki gösterildiği gibi konumlanır.

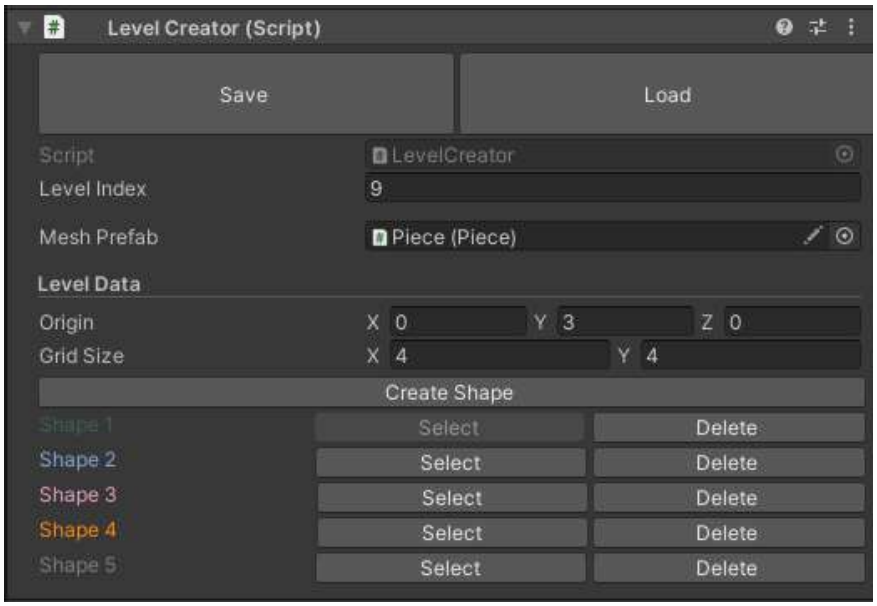


LevelData'lar server'dan kolay çekilebilmesi için Json formatında saklanmaktadır.

LevelCreator ve LevelEditor

Proje içerisinde level oluşturmayı ve düzenlemeyi kolaylaştırmak amacı ile LevelEditor oluşturulmuştur.

LevelEditor, LevelCreator sahnesinde yer alır.

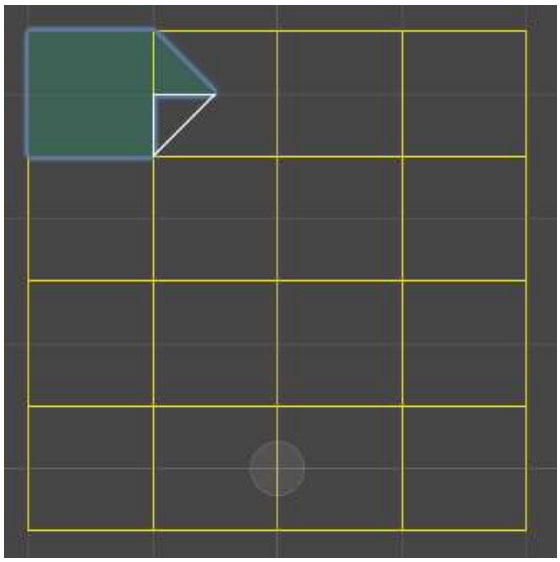


Yukarıda görüldüğü gibi bir arayüze sahiptir. Origin ve GridSize değerleri değiştirildiğinde, oluşacak olarak grid sahnede anlık olarak görüntülenebilir.

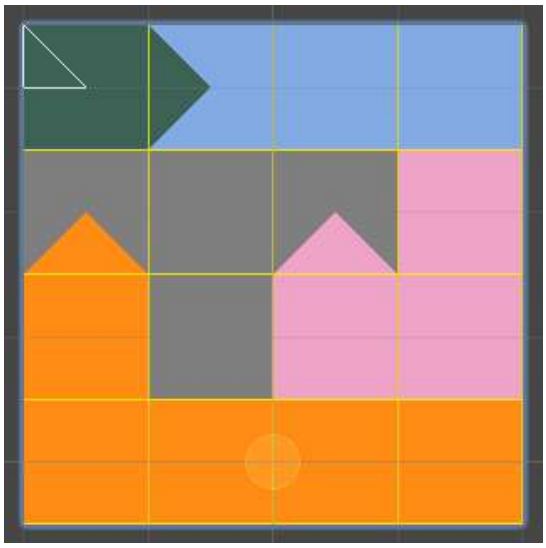
Shape Oluşturma
"Create Shape" butonuna basılarak yeni bir shape datası oluşturulabilir. Save butonunun aktif olması için en az beş shape oluşturulmuş olmalıdır. On iki shape oluşturulduğunda Create Shape butonu pasif konuma geçer.

Oluşan Shape'in inspector'daki label'ı sahnedeki rengi ile aynı olacaktır. Bu sayede seçilen shape'i sahnede tespit etmek kolaylaşır.

Select butonundan bir shape seçildiğinde scene üzerinden mouse'a basılı tutarak grid üzerinde istenildiği gibi şekillendirilebilir. Mouse'un tutulduğu yere en yakın tri parlayarak seçili olan tri'yi gösterir.



Bütün grid oluşturulan shape'ler ile boyandığında Save butonuna basılan seviye istenilen LevelIndex ile kaydedilebilir.



Ayrıca level index girildikten sonra Load butonuna basılarak önceden oluşturulmuş olan LevelData sahneye yüklenebilir ve üzerinde değişiklik yapılabilir. Değişikliği kaydetmek için aynı level index ile kaydetmek yeterli olacaktır.

Load edilen level index'i mevcut değil ise 4,4'e grid boyutlarında boş bir level oluşturulurur.

Diğer Yapılar

GameState

Play ve Level Completed state'leri arasında geçiş yapabilmek için basit bir State Pattern uygulanmıştır.

Camera

Kamera grid boyutuna göre otomatik olarak konumlanır.

Dynamic Spawn Zone

Piece Spawner'ın spawn zone'u kamera konumuna ve grid konumuna bağlı olarak dinamik olarak belirlenir.