Tetris Tipinde Kelime Oyunu

Ali Kutay Bilgilioğlu 200202108@kocaeli.edu.tr Sezer Yildirim 210202001@kocaeli.edu.tr

Nisan 2023

1 Özet

Bu uygulamada tetris yapısına benzer bir kelime oyunu tasarlanmıştır. Mobil programlama esasına dayalı bir mimari istendiğinden uygulama Flutter'da Dart dili ile yazılmıştır. Flutter'ın bir native dil olması mobil programlamada fazlaca kullanılmasına sebep olmuştur. Uygulamadaki gelişmeleri gözlemlemek, oyunu oynayabilmek için Android Studio Virtual Device Manager ile emülatör kullanılmıştır. Çalışmaya öncesinde React Native ile başlanılmış ancak dilin çevre değişkenlerinin kurulumu ve bu değişkenlere bağlı hatalarda fazlaca tıkanılmıştır. Flutter da ise tam tersi bir yapı gözlemlenmiştir. Bunun sebebi flutter'ın flutter doctor özelliğinin olmasıdır. Uygulama geliştirme ortamı olarak Visual Studio Code kullanılmıştır.

2 Giriş

Oyun 8x10'luk(yani 8 sütun 10 satır olacak) bir alan üzerinde harfler yukarıdan aşağı gelecek şekilde ilerlemelidir. En üst kısmından aşağıya doğru birim birim ilerlemektedir. Uygulamada asenkron olarak çalışması beklenen yapılar olduğundan asenkron programlama için gereken sınıfları içerir. Bu fonksiyonlar dart:async içerisinde gerçekleştirilmiştir. Kullanıcı arayüzü elemanlarını ve materyal tasarım stillerini içeren Flutter'ın temel paketi olarak material.dart kullanılmıştır. Oyun ilk başladığında yukarıdan aşağıya doğru 3 satır tamamen dolu olacak şekilde rastgele harfler indirilmesi gerekmektedir. Başlangıç aşaması sonrasında ise harflerin rastgele ilk satırın tek bir sütununda ilerlemesi ve düşmesi gerekmektedir. Bu yapının sağlanması için setState ve initState arasında bir ilişki kurulmuştur. Buna göre oyun ilk başladığı anda 3 satır tüm sütunlarında rastgele olacak şekilde dolmaktadır. Bu sütunlar dolduktan sonra initstate düzenli olarak rastgele harf üretmeye ve bunları kaydırmaya devam etmektedir.

```
  void initState()
super.initState();
for ( starteri=0 ; starteri ¡ 3; starteri++)
generateRandomLetter();
for ( starterj=0 ; starterj ¡ 9; starterj++)
shiftLetterDown();


Timer.periodic(Duration(seconds: 5), (timer)
=¿
singgenerateRandomLetter());
.
.
.
Timer.periodic(Duration(milliseconds: 500),
(timer) =¿
singshiftLetterDown());
```

3 Program Mimarisi

İlk olarak, gerekli kütüphaneler import edilir: dart:async, package:flutter/material, dart:math, dart:convert ve package:flutter/services.

Daha sonra, main() fonksiyonu ile uygulama başlatılır ve MyApp sınıfı, StatelessWidget sınıfından kalıtım alarak oluşturulur. Bu sınıf, MaterialApp widget'ını kullanarak uygulamanın altyapısını belirler.

MyHomePage sınıfı ise StatefulWidget sınıfından kalıtım alır ve oyunun oynanacağı ana sayfa widget'ını tanımlar.

_MyHomePageState sınıfı, MyHomePage sınıfının özel bir durum
sınıfıdır ve StatefulWidget'in durumunu tutar. Bu sınıf, kullanıcının
puanını, yüksek puanını, check butonuna basıldığında hata yapma sayıcısını,
türkçe harfler için manuel olarak oluşturulmuş diziyi, en yüksek skor gibi değişkenleri tanımlar ve oyunun işleyişinde temel olan tüm fonksiyonlar burada tanımlanır.

Son olarak, trletters listesi, Türk alfabesi harflerini içerir ve oyun sırasında kullanılacak rastgele harfleri oluşturmak için kullanılır. Daha önce ASCII kodları üzerinden rastgele seçilen harfler türkçeyi karşılaması amacıyla manuel olarak bir liste oluşturularak rastgelelik sağlanmıştır. Bu sayede türkçe karakterlerden rastgele akan bir harf matrisi oluşur.

final List¡String¿ trletters = ['A', 'B', 'C', 'Ç', 'D'....
.
.
.
letter = trletters[random.nextInt(29)];
scores listesi, daha önceki oyunların puanlarını tutar ve highScore değişkeni, oyun boyunca kullanıcının elde ettiği en yüksek puanı saklar.

İlk aşamada matris yapısının oluşturulması gerekmektedir. Bu matris yapısı direkt olarak bir String matrisi olarak oluşturulmuştur. Matris önce karelere parsellenip sonrasında harfler yerleştirilebilirdi. Buradaki amaç String matrisi oluşturarak birer birer kayma işlemini ve seçilebilme işlemini sağlayabilmektir. String yapısı sayesinde önce Stringler boş olarak atanıp sonrasında rastgele olarak boş olan matris değişmektedir. Bu şekilde kayma var gibi görülmektedir.

String letter = ''; satırı, bir karakter dizisi değişkeni olan letter'ı tanımlıyor ve boş bir değerle başlatıyor. List_List_String__ letters = List.generate(10, (index) => List.generate(8, (index) => '')); satırı, letters isimli iki boyutlu bir liste oluşturuyor. Liste, 10 satır ve 8 sütundan oluşuyor. Her bir hücre boş bir karakter dizisiyle başlatılıyor. int currentRow = 0; ve int currentCol = 0; satırları, currentRow ve currentCol adında iki tamsayı değişkeni tanımlıyor. Bu değişkenler, kullanıcının hangi hücrede olduğunu takip etmek için kullanılıyor. List¡String¿ selectedLetters = []; ve List<List<int>> selectedIndexes = []; satırları, sırasıyla selectedLetters ve selectedIndexes adında iki liste oluşturuyor. Bu listeler, kullanıcının seçtiği harfleri ve harflerin konumunu saklamak için kullanılıyor. void selectLetter(int row, int col) fonksiyonu, row ve col parametreleri ile çağrıldığında, belirtilen konumdaki hücredeki harfi seçiyor. Fonksiyon, seçilen harfi selectedLetters listesine ve harfin konumunu selectedIndexes listesine ekliyor. void clearSelectedLetters() fonksiyonu, selectedLetters listesini temizliyor. void checkcounter() fonksiyonu, rastgele bir harf üretiyor ve shiftLetterDown() fonksiyonunu çağırarak harflerin konumlarını aşağı kaydırıyor. Timer sınıfı kullanılarak, her 0.5 saniyede bir bu işlem tekrarlanıyor.

Kelime havuzunun oluşturulması için TDK reposu kullanılmıştır. Ancak bu yapının filtrelenmesi gerekmektedir.
Kelime sayısı 50.000 den fazla kelime içermelidir.
Havuzdaki kelimeler en az 3 harfli kelime

olmalıdır.
Havuzdaki kelimeler tek bir kelime şeklinde olmalıdır. Birden fazla kelime içeren yapılar bulunmayacaktır.

Bu kelime filtrelemelerinin olduğu bir json dosyası hazırlanmıştır. Json dosyası proje klasörünün altına eklenerek bu dosyaların decode edilmesi sağlanmış ve bir checkword fonksiyonu ile seçilen harfler karşılaştırılmıştır.

Future¡List¡String¿¿ getWords() async
String jsonString = await rootBundle.loadString('words.json');
List¡dynamic¿ wordsJson = json.decode(jsonString)["words"];
List¡String¿ words = wordsJson.cast¡String¿();
return words;

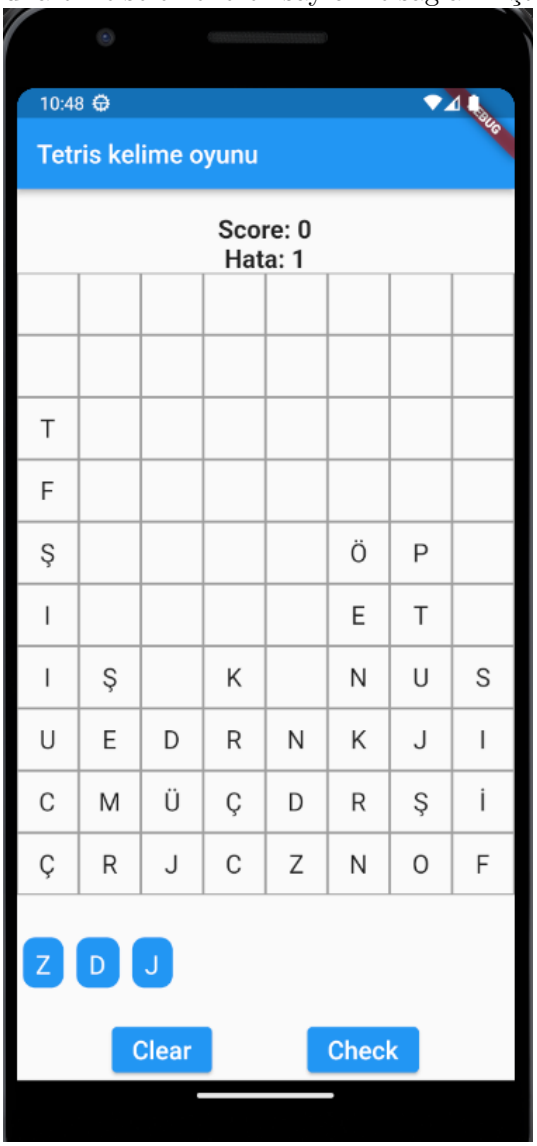
void checkWord() async
String word = selectedLetters.join('');
List¡String¿ words = await getWords();
if (words.contains(word.toLowerCase()))

Burada fonksiyonların async anahtar kelimesine dikkat edilmelidir.Bu fonksiyonun async olarak işaretlenmesinin sebebi, getWords() fonksiyonunun asenkron olarak çalışmasıdır. await anahtar kelimesi ile getWords() fonksiyonunun tamamlanmasını beklediği için, checkWord() fonksiyonu da asenkron olarak çalışır.

Eğer getWords() fonksiyonu senkron olarak çalıştırılmak istenseydi, getWords() fonksiyonu ağ istekleri veya veritabanı işlemleri gibi yavaş işlemler içerebilir, bu durumda senkron olarak çalıştırıldığında uygulamanın donmasına veya yavaşlamasına neden olabilir. Kelime listesi 120 bin satırlık bir yapı olduğundan bu işlemi asenkron yapmak önemlidir. Ancak bu kodda veritabanı işlemi yoktur. Ayrıca ağ isteği olarak da localhost olduğundan senkron da kullanılabilirdi.

Oyundaki puanlandırma sistemi için bazı özel koşullar vardır. Bu özel koşullarda checkword yapısının içerisinde doğru olma durumuna eklenerek puanlama sistemi yapılmıştır.

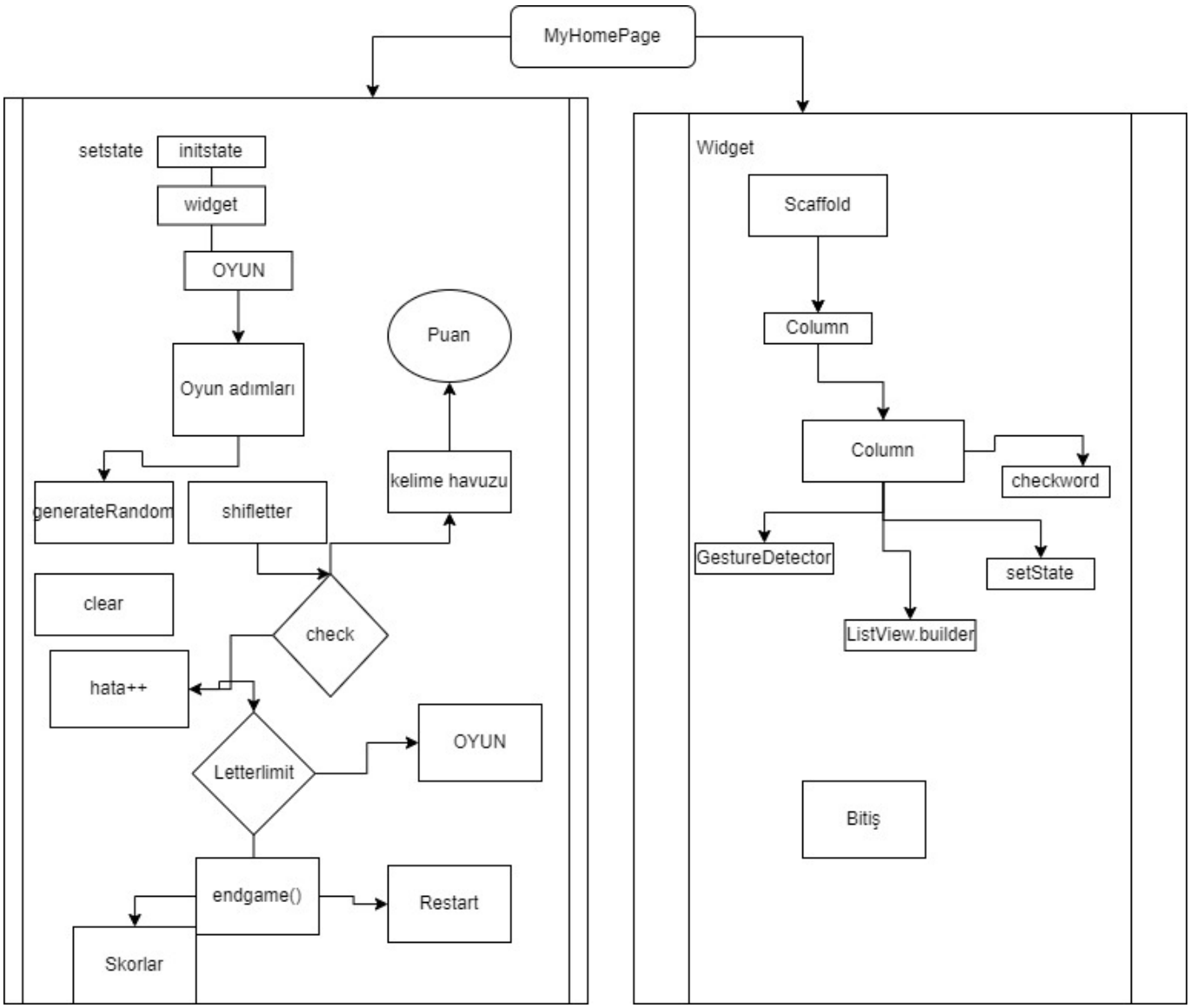
Aynı zamanda 3 kere yanlış kelime girilmesi durumunda ilk satır tüm sütunlarında rastgele kelime girilerek bu kelimelerin aşağı düşmesi sağlanmıştır. Bu sayıcı da checkword fonksiyonuna bir else eklenerek sayıcı ile sağlanmıştır.



4 Referanslar

1) github.com (05.04.2023)
2)https://stackoverflow.com/(12.04.2023)
3)https://flutter.dev/(19.04.2023)
4)https://github.com/(22.04.2023)
5)https://tutorials.point.com/flutter(07.04.2023)

5 Akış


MyHomePage
setstate
initstate
widget
OYUN
Oyun adımları
Puan
kelime havuzu
generateRandom
shifletter
clear
check
hata++
Letterlimit
OYUN
endgame()
Restart
Skorlar
Widget
Scaffold
Column
Column
checkword
GestureDetector
setState
ListView.builder
Bitiş