Dart Notlarım **02.09.2024**

- ~/ operatörü bölmeyi yap alt değerini ata. Misal 5~/2 sadece 2 cevabını verir.

- final anahtar kelimesi, ilk değer atamadan sonra(runtime'da atanan o anki değeri alır, sayfa açılırken bu şekilde bir değer atanıp sabitlenebilir) değişkenin değerini değiştirilemez hale getirir.

- const anahtar kelimesi, proje derlendiği andan itibaren değeri sabitler ve değiştirilemez hale getirir.

- Listeler için bu anahtar kelimelerin kullanımında bazı istisnalar mevcuttur.

```
void main(){
//listenin elemanları değiştirilebilir (mutably) ancak listenin kendisi (referans) değiştirilemez.
 final List<int> liste1 = [10,20,30,40];
 liste1.add(50);

//const anahtar kelimesi listeyi ve içeriğini derleme zamanında sabitler ve listenin içeriği değiştirilemez.
//Derleme zamanında hata verir.
 const List<int> liste2 = [10,20,30,40];
 liste2.add(50);
}
```

- Bazı liste fonksiyonlarının örnekleri.

```
void main(){
// İstenilen: 100 tane musteri oluştur bunlarin hepsine sira ile numarasina gore 5 tl ekle

//List.generate fonksiyonu bir döngü misali çalışır ve kendisine verilen parametrelere ve fonksiyon içerisindeki direktiflere göre her dönüşünde 1 eleman yaratır.
List<double> customersMoney = List.generate(100, (index){
  //index değerini int olarak alır ve üzerine 5 ekler.
  return index + 5;
});

print(customersMoney);
customersMoney.add(1.0);//liste sonuna verilen değeri ekler.
customersMoney.insert(0, 99.9);//0. indexteki değeri 99.9 ile değiştirir.
customersMoney.sort();//Elemanları küçükten büyüğe sıralar.
print(customersMoney.length);//Listenin uzunluğunu döndürür.
print(customersMoney.reversed);//Ters çevirir.
print(customersMoney.isEmpty);//Liste boş mu? Boolean değer döndürür.
print(customersMoney.last);//Son elemanı döndürür.
customersMoney.contains(5.0);// Listede 5.0 var mı?
customersMoney.clear();//Liste elemanlarını temizler.
print(customersMoney);
}
```

- Fonksiyonlar, parametreler, opsiyonel parametreler, required

```
void main(){
    final    newResult    =    convertToStandartDolar(100,
dolarIndex: 13);
 final newResult2 = convertToStandartDolar(100);
 final newREsult3 = convertToEuro(userMoney: 500);
}

int convertToDolar(int userMoney) {
 return userMoney ~/ 13;
}

//Opsiyonel parametreler sayesinde fonksiyon çağrısında
ilgili parametreye değer yollanmaz ise o parametrenin
default değeri kullanılır.
//Değer  yollandığı  takdirde  default  değer  dikkate
alınmaz.
int convertToStandartDolar(int userMoney, {int dolarIndex
= 14}) {
 return userMoney ~/ dolarIndex;
}

int convertToEuro({required int userMoney, int dolarIndex
= 14}) {
 return userMoney ~/ dolarIndex;
}
```

- Map yapısı

```
void main() {
 //Müşteriler
 Map<String, int> users = {'ahmet': 20, 'mehet': 30};

 //Müşteri Ahmet'in ne kadar parası var?
 print('ahmetin parasi ${users['ahmet']}');

 //Tüm müşterileri user-value şeklinde tek tek print ile
gösterme.
 for (var item in users.keys) {
   print('${item} - ${users[item]}');
 }


 print('----------------');
```

```
 for (var i = 0; i < users.length; i++) {
                  print('${users.keys.elementAt(i)}      -
${users.values.elementAt(i)}');
 }

  //Bankanın müşterileri ve müşterilerin birden fazla
hesabı olabilir.
  //Hesapları kontrol et ve herhangi bir hesapta olan
tutar 150'den büyük ise krediyi onayla.
 print('-----------------');
 final Map<String, List<int>> vbBank = {
   'ahmet': [100, 300, 200]
 };
 vbBank['mehmet'] = [30, 50];
 vbBank['veli'] = [30];

 for (var item in vbBank.keys) {
   //Bankanın tüm elemanları

   for (var money in vbBank[item]!) {
     //User'ın hesaplarını dolaşma
     if (money > 150) {
       print('kredin hazir');
       break;
     }
   }
 }

 //Her müşterinin toplam bakiyesi.
 for (var name in vbBank.keys) {
   //vbBank[item]!-> Müşterinin hesapları demek
   int result = 0;
   for (var money in vbBank[name]!) {
     result = result + money;
   }

   print('$name senin toplam paran -> $result');
 }
}
```

- ?, ! işlemleri

```
void main() {
 // ? bir değerin nullable olabileceğini ifade eder.
 int? newMoney;

// ! işareti bir işlemi forcelama/zorlama anlamına gelir. Bilinçsiz kullanıldığı takdirde bol bol crashe sebep olur.
//print(newMoney! + 10);

//  Beklenmeyen değer gelme ihtimaline karşın ilgili durumun bir şekilde handle edilmesi gereklidir.
 if (newMoney != null) {
   print(newMoney + 50);
 } else {
   print(10 + 10);
 }
}
```

- Listeler gerçek senaryolarda null değer içerebilir.

```
void main() {
//  bankaya  3  tane  musteri  gelir  birinin  100tlsi  var digerinin hesabi hic yok digerinin 0 tlsi var
//  hesabi  olmayana  hesap  acalim,  0  tlsi  olani  kov, 100tlsi olana musterim hosgeldin

 List<int?> customerMoneys = [100, null, 0];

 for (var item in customerMoneys) {
   if (item != null) {
     if (item > 0) {
       print('Müşterim hoşgeldin');
     } else {
       print('Maalesef size hesap açamıyoruz.');
     }
   } else {
     print('Haydi hesap açalım.');
   }
 }
}
```

- Fonksiyonlarda null değer döndürebilir.

```
int? controlMoney(int? money) {
 if (money != null && money > 0) {
   return money;
 }else{
   return null;
 }
  //Burada else bloğunda null döndürülmese de dart
otomatik olarak null döndürür.
}
```