Android Mobil Uygulama Geliştirme Eğitimi | Kotlin

Değişkenler ve Veri Tipleri

Kasım ADALAN
Elektronik ve Haberleşme Mühendisi
Android - IOS Developer and Trainer

Eğitim İçeriği

1. Değişkenler
2. print() metodu
3. Constant – Sabitler
4. Aritmetik Operatörler
5. Tür Dönüşümü
6. Konsol Girdisi

Variables - Değişkenler

Değişkenler

- Modern diller hafızada saklanan değerleri değişkenler ile ifade etmektedir.
- Değişkenler hafızada geçici olarak saklanan değerleri temsil eder .
- Kotlin dilinde değişken için tür belirtmemize gerek yoktur.

Not : Değişkenler kalıcı değildir.
Programdan çıkıldığında değerler kaybolur. Kalıcı değerler için değişkenlerin değerleri diske yazılmalıdır.

Artık ; yok

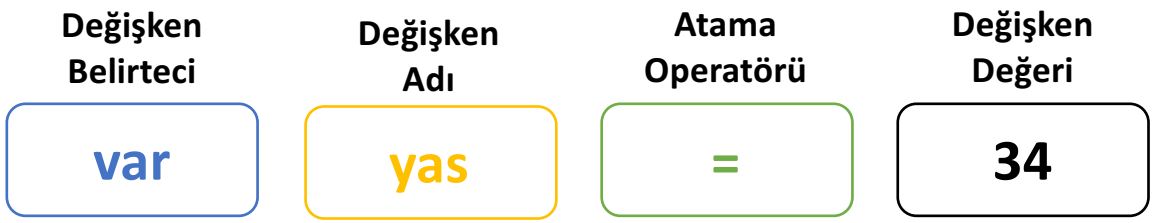
- Modern bir dil olan kotlin için kod satırı bittikten sonra ; koyulmasına gerek yoktur.

- El alışkanılığı ile ; koysanız bile problem olmaz hata almazsınız.

- İstisna :
- Eğer tek satırda iki farklı kod yazmak istersek mecburen ; koymalıyız.

```
var yas : Int = 34 ; print(yas)
```

Değişken oluşturma




```
var yas  = 34
```

Tür Belirterek Değişken oluşturma


```
var yas : Int = 34
```

Data Tipleri

Tam Sayılar

Long
Int
Short
Byte

Ondalıklı Sayılar

Double
Float

Metinsel İfadeler

String : Yazılar
Char : Harfler

Mantıksal İfadeler

Boolean : True veya False

Data Boyutları

Type	Size
Double	64
Float	32
Long	64
Int	32
Short	16
Byte	8

Bit türündedirler

Literals – Değerlerin Yazılma Kuralları

- Literals değişkenler için kullanılan değerlerin nasıl yazılması gerektiğini temsil eder.

```
"Ahmet" //String ( Metinsel İfade )
'a' //Char ( Harfsel İfade )
18 // Tam sayı
1.78 // Double ( Ondalıklı Sayı )
1.78f // Float ( Ondalıklı Sayı )
```

Değişkenlere isim verme kuralları

- Case sensitive'dir.Büyük küçük harf farkı vardır.
- Rakamla başlayamaz.
- @, $, ve % değişken içerisinde kullanılmaz.

- Bazı örnekler ;

```
Azad          zara    abc     move_name     a_123
myname50      _temp   j       a23b9         retVal
```

Örnek 1:

- Bir öğrencinin adını ,yaşını ,boyunu ve adının baş harfininin tutulduğu değişken oluşturunuz.

Örnek 2:

- Bir şirketin ürünlerinin bilgilerinin tutulduğu ürünler tablosunu temsil eden değişkenleri oluşturunuz.

ürün_id	ürün_adi	ürün_adet	ürün_fiyat	ürün_tedarikci
3416	Kol saati	100	149.99	rolex

print() ve println() metodu ile Çıktı Alma

- Bu metodları kodlama yaparken sıkça kullanırız.
- Kodlama yaparken kodların çalışma sonuçlarını bu metod ile takip edebiliriz.
- print() yan yana , println() alt alta yazmak için kullanılır.

```
print("Deneme1")
println("Deneme2")
print("Deneme3")
```

```
Deneme1Deneme2
Deneme3
```

Değişkenleri Yazdırma

String ifade içine $ ifadesi kullanılarak çıktıya değişken eklenebilir.

```
var ad = "Ahmet"
var yas = 10

println("$ad Bursada $yas yıldır yaşamaktadır.")
```

Ahmet Bursada 10 yıldır yaşamaktadır.

Değişkenleri Yazdırma

String ifade içine ${} ifadesi kullanılarak işlem yapılabilir.

```
var a = 10
var b = 20

println("$a ve $b nin toplamı : ${a + b} dir.")
```

10 ve 20 nin toplamı : 30 dir.

Değişken Oluşturma Çeşitleri

```
var sayi = 10

var s1 = 80
var s2 = 70

var toplam = s1 + s2
```

Kullanılamayan çeşitler

~~`var sayi1 = 30 ,sayi2 = 40, kelime = "merhaba"`~~

~~`var sayi`~~ **Primitif tiplerin varsayılan başlangıç değeri olmak zorunda**

~~`lateinit var k`~~

Type Safety – Tür Güvenliği

- Oluşturduğunuz değişkene farklı türde değişken atayamazsınız.

```
var varA = 42
varA = "This is hello"
print(varA)
```

Değişkenin kapsamı (Global ve Local Değişken)

- Süslü parantez **{ }** bizim kapsamımızı belirler. Değişkenin ulaşılabilirliği buna bağlıdır.

```
class Deneme {
    var x = 10 //Global Değişken
    var y = 20 //Global Değişken

    fun  topla(){
        var x = 40 //Local Değişken
        x = x + y //Burda x lokal y global değişkendir.
        //lokal değişken global değişkene baskın gelmiş
        // ve lokal değişken geçerlidir.
        print(x)
    }


}
```

Constant - Sabitler

Constant - Sabitler

- Sabitler içerisine bir kere veri atıldığında bir daha değiştiremeyeceğiniz yapılardır.
- **val** ismi ile kullanılırlar.
- val kullanmak memory yönetimini rahatlatır.
- Çünkü hafızada sabit için yer ayrılır ve değişim olmayacağı için açılan yer yeni bir değer almak için beklemez.
- Sadece kullanılma amaçlı değişkenler için kullanılması için uygundur.
- Özellikle nesne tabanlı programlamada kullanılır.

```
val pi = 3.14

pi = 3 //let olan değişkene daha sonra değer atanmaz.

val klorOrani:Double = 4.5

val isim = "Ahmet"
```

Kaçış Karakterleri

- Kaçış karakterleri String ifade içine bazı karakterleri yazmamızı sağlarlar.
- Bunun çıkış sebebi String ifadelerin ” işareti ile başlayıp bitmesidir.

```
var varA = "Godzilla"
```

- En çok kullanılan kaçış karakterleri.
  - **** – \ işareti
  - **\t** – Bir tab boşluk bırakır
  - **\n** – Bir alt satıra iner
  - **\”** – Çift tırnak işareti
  - **\’** – Tek tırnak işareti

```
var stringL = "Hello\tWorld\n\nHello\'Swift 4\'"
print(stringL)
```

```
Hello World


Hello'Swift 4'
```

Örnek

Merhaba bu ”android"
eğitiminde \kotlin\ dilini öğrenicez

Yorum Satırları

- Yorum satırı kullanımının birçok amacı vardır.
- Kodunuza anlaşılır notlar yazmak.
- Bazı kod satırını geçici olarak gizlemek için kullanılabilir.

- Satıra yorum ekleme ;

  // işareti ile yapılır.

  Örn : // Bu nesne ile veri tabanına erişelebilir.

- Blok yorumu ekleme;

  /* ile açılır */ kapatılır.Tek satır değil birden fazla satır için kullanılabilir.

  Örn : /* Açıklama
  Veri tabanı için gerekli nesneleri kullanmalıyız.
  Bazı nesneler nil dönebilir dikkatli olunmalıdır. */

Aritmetik Operatörler

- Matematiksel işlemleri yapmamızı sağlarlar.
- Parantezler işlemin önceliğini belirtmek için kullanılır.
- Örn : A = 10 ve B = 20 olsun

Operator	Açıklama	Örnek
+	Toplama	A + B = 30
−	Çıkarma	A − B = -10
*	Çarpma	A * B = 200
/	Bölme	B / A = 2
%	Mod İşlemi	B % A = 0

Örnekler : Aşağıdaki formülleri tanımlayınız.

- **Daire alanını değişkenler oluşturarak hesaplayınız.**

- **F = m x a Uygulanan Kuvvet(F)= Cismin kütlesi(m) x cismin ivmesi (a)**

$$\Delta x = (\frac{v + v_0}{2})t$$

$$\Delta x = v_0 t + \frac{1}{2}at^2$$

Atama Operatörlerinin Kısaltımı

- Atama işlemlerini kolaylaştırma amaçlı kullanılırlar.
- Aritmetik operatörlerin hepsinde geçerlidir.
- Normal ifade ;
- a = a + 3
- **Kısayol** : a +=3

b = b * 3
Kısayol : b *=3

c = c - 3
Kısayol : c -=3

d = d / 3
Kısayol : d /=3

```
var  y = 10

y = y + 2

y+=2

print(y)//14
```

Tür Dönüşümü

1. Sayısaldan sayısala dönüşüm

2. Sayısaldan metine dönüşüm

3. Metinden sayısala dönüşüm

- **toDouble() , toFloat() , toLong() , toInt() , toShort() , toByte() , toChar() , toString()**

Sayısaldan sayısala dönüşüm

```
var i:Int = 42
var d:Double = 42.45
var f:Float = 42.89f

var sonuc1:Int = d.toInt()
var sonuc1:Int = f.toInt()
var sonuc1:Float = i.toFloat()
```

Sayısaldan Metine Dönüşüm

```
var sayi1:Int = 42
var sayi2:Double = 42.45
var sayi3:Float = 42.89f

var srt1 = sayi1.toString()
var str2 = sayi2.toString()
var str3 = sayi3.toString()
```

Metinden Sayısala Dönüşüm

- Dönüşüm olurken dikkatli olunmalıdır çünkü metin içinde her zaman sayı yer almaz hata ihtimali yüksektir.

YÖNTEM 1

```
var str = "48T"

try {

  val sayi = str.toInt()

}catch(nfe:NumberFormatException){
  //Dönüşümde sorun var
}
```

Metinden Sayısala Dönüşüm

- Dönüşüm olurken dikkatli olunmalıdır çünkü metin içinde her zaman sayı yer almaz hata ihtimali yüksektir.

YÖNTEM 2

```
var str = "48T"

val sayi = str.toIntorNull()

if (sayi =! null){
  //hata yok işlem yapılabilir
}else{
  //dönüşüm hatası var
}
```

Metinden Sayısala Dönüşüm

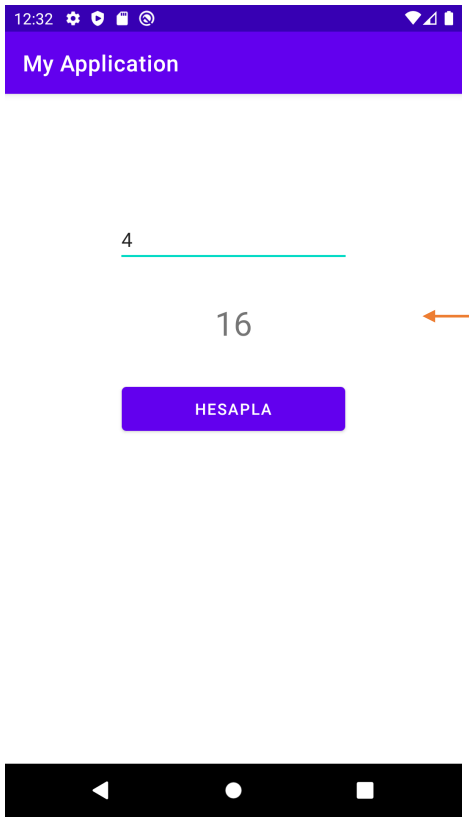
- Dönüşüm olurken dikkatli olunmalıdır çünkü metin içinde her zaman sayı yer almaz hata ihtimali yüksektir.
  - Safe call

YÖNTEM 3

```
var str = "48T"

val sayi = str.toIntorNull()

sayi?.let {
  //Dönüşümde sorun yoktur
}
```

Android Kullanım Alanı

1. Durum

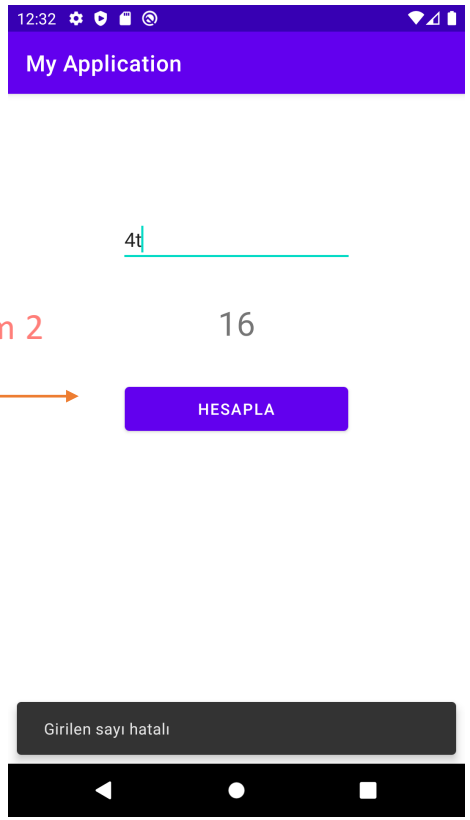



```
buttonHesapla.setOnClickListener {  it: View!

    val gelenVeri = editTextGirdi.text.toString()

    val sayi = gelenVeri.toIntOrNull()//String to Int

    if(sayi != null){
        val sonuc = sayi * sayi
        textViewCikti.text = sonuc.toString()//Int to String
    }else{
        Snackbar.make(buttonHesapla
            , text: "Girilen sayı hatalı",Snackbar.LENGTH_SHORT).show()
    }

}
```

Dönüşüm 1

Dönüşüm 2

2. Durum



Konsol Girdisi

```
println("Adınızı giriniz")

val girdi = Scanner(System.`in`)

val ad = girdi.next()

println("Adınız : $ad")
```

Tüm türlerde girdi olabilir.

Bunu sağlamak için girdiye uygun tür metodu seçilmelidir.

Örn : int -> nextInt() , double -> nextDouble() vb.

Teşekkürler...


kasım-adalan


kasimadalan@gmail.com


kasimadalan