

# Swift OOP Eğitimi

## Değişkenler ve Veri Tipleri

Kasım ADALAN

Elektronik ve Haberleşme Mühendisi  
Freelance Software Developer

# Eğitim İçeriği

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

Variables - Değişkenler

# Değişkenler

- Modern diller hafızada saklanan değerleri değişkenler ile ifade etmektedir.
- Değişkenler hafızada geçici olarak saklanan değerleri temsil eder .
- Swift 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 swift 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

Değişken Belirteci	Değişken Adı	Atama Operatörü	Değişken Değeri
var	yas	=	34

```
var yas = 34
```

## Tür Belirterek Değişken oluşturma

Değişken Belirteci	Değişken Adı	Tür Belirteci	Değişken Türü	Atama Operatörü	Değişken Değeri
var	yas	:	Int	=	34

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

# Data Tipleri

## Tam Sayılar

***Int*** : 0 dahil Pozitif ve Negatif Sayılar

***UInt*** : 0 dahil Pozitif Sayılar

## Ondalıklı Sayılar

Double

Float

## Metinsel İfadeler

String : Yazılar

Character : Harfler

## Mantıksal İfadeler

Bool : True veya False



# Data Boyutları

Type	Typical Bit Width	Typical Range
Int8	1byte	-127 to 127
UInt8	1byte	0 to 255
Int32	4bytes	-2147483648 to 2147483647
UInt32	4bytes	0 to 4294967295
Int64	8bytes	-9223372036854775808 to 9223372036854775807
UInt64	8bytes	0 to 18446744073709551615
Float	4bytes	1.2E-38 to 3.4E+38 (~6 digits)
Double	8bytes	2.3E-308 to 1.7E+308 (~15 digits)

**Int** 32 bit telefonlarda **Int32** yerine geçer.

**Int** 64 bit telefonlarda **Int64** yerine geçer.

# Literals – Değerlerin Yazılma Kuralları

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

```
"Ahmet" //String ( Metinsel ifade )  
18       //Int ( Tamsayı ifade )  
1.78     //Double ( Ondalıkli sayı ifade )  
"A"      //Character ( Harfsel ifade )
```

# 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

## White Space – Beyaz Boşluk

- Kodlama yaparken daha düzenli görünmesi için Swift boşluk bırakmamızı istemektedir.

```
var meyve = elma + kiraz // doğru kullanım  
var meyve= elma + kiraz // yanlış kullanım  
var meyve =elma + kiraz // yanlış kullanım  
var meyve = elma+ kiraz // yanlış kullanım  
var meyve = elma +kiraz // yanlış kullanım
```

## Örnek 1:

- Bir öğrencinin **adini** , **yaşını** , **boyunu** ve **adının baş harfinin** tutulduğu değişkenler 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() metodu ile Çıktı Alma

- print() metodu kodlama yaparken sıkça kullandığımız bir yapıdır.
- Kodlama yaparken kodların çalışma sonuçlarını bu metod ile takip edebiliriz.
- Diğer diller gibi print ve println ayrımı yoktur.
- Varsayılan olarak println gibi çalışır.Yani alt alta yazar.

```
print("Merhaba") //Merhaba
```

```
print(1.0,2.0,3.0) // 1.0 2.0 3.0
```

```
print(1.0,2.0,3.0, separator: "-")//1.0-2.0-3.0
```

Separator ifadelerin aralarına istenilen ifadeyi yerleştirmemize yarar.

# Değişkenleri Yazdırma

String ifade içine `\()` ifadesi kullanılarak çıktıya değişken eklenebilir.

```
var ad = "Ahmet"  
var yıl = 10
```

```
print("\(ad) Bursada \(\yıl) yıldır yaşamaktadır.")
```

**Ahmet Bursada 10 yıldır yaşamaktadır.**



# Değişken Oluşturma Çeşitleri

```
var sayi = 10
```

```
var sayi1 = 30 ,sayi2 = 40, kelime = "merhaba",harf = "f"
```

```
var s1 = 80
```

```
var s2 = 70
```

```
var toplam = s1 + s2
```

```
var numara:Int?
```

```
numara = 20
```

# Type Safety – Tür Güvenliği

- Oluşturduğunuz değişkene farklı türde değişken atayamazsınız.

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

```
main.swift:2:8: error: cannot assign value of type 'String' to type 'Int'  
varA = "This is hello"
```

# 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  
  
    func 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.
- **let** ismi ile kullanılırlar.
- let 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.

```
let pi = 3.14
```

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

```
let klorOrani:Double = 4.5
```

```
let isim = "Ahmet"
```

# Kaçış Karakterleri

- Kaçış karakterleri String ifade içine bazı karakterleri yazmamızı sağlarlar.
- Bunun çıkışı 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

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

```
Hello World  
  
Hello'Swift 4'
```

# Örnek

Merhaba bu "ios"  
eğitiminde \swift' dilini öğrenecezs

# 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 erişilebilir.

- **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.
- $F = m \times a$  Uygulanan Kuvvet(F)= Cismin kütlesi(m) x cismin ivmesi (a)

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

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

# Atama Operatörlerinin Kısaltımı

- Atama işlemlerini kolaylaştırma amaçlı kullanılırlar.
- Aritmetik operatörlerin hepsinde geçerlidir.
- Normal ifade ;

•  $a = a + 3$

$b = b * 3$

$c = c - 3$

$d = d / 3$

• **Kısayol** :  $a += 3$

**Kısayol** :  $b *= 3$

**Kısayol** :  $c -= 3$

**Kısayol** :  $d /= 3$

```
var y = 10
```

```
y = y + 2
```

```
y+=2
```

```
print(y)//14
```

# Tür Dönüşümü

1. Sayısalardan sayısal dönüşüm
  2. Sayısalardan metne dönüşüm
  3. Metinden sayısal dönüşüm
- **Int() , Float() , Double() , String()**

# Sayısalardan sayısal dönüşüm

```
//Sayısalardan Sayısal
```

```
var i:Int = 42
```

```
var d:Double = 42.45
```

```
var f:Float = 42.89
```

```
var sonuc1:Int = Int(d) //42 Double to Int
```

```
var sonuc2:Double = Double(i)//42.0 Int to Double
```

```
var sonuc3:Int = Int(f)//42 Float to Int
```

```
var sonuc4:Float = Float(i)//42.0 Int to Float
```

## Sayısalardan Metne Dönüşüm

```
var sayi1 : Int = 42
var sayi2 : Double = 42.45
var sayi3 : Float = 41.89

var str1 = String(sayi1)
var str2 = String(sayi2)
var str3 = String(sayi3)
```

# Metinden Sayısala Dönüşüm

- Dönüşüm olurken unwrapping olmalıdır çünkü metin içinde her zaman sayı yer almaz hata ihtimali yüksektir.

```
var str = "37"

if let sayi = Int(str) {
    print(sayi)
}

var str1 = "37.56"

if let sayi1 = Double(str1) {
    print(sayi1)
}

if let sayi2 = Float(str1) {
    print(sayi2)
}
```

# Tuples



# Tuples

- Tuples class ve struct yapılarının basit halidir.
- Farklı türde verileri içerisinde tutabilir.
- Çoklu değişken gibidir.

```
var kisi = ("Ahmet", "Aksoy")
```

```
var ad = kisi.0 // Ahmet
```

```
var soyad = kisi.1 // Aksoy
```

# Veri Okuma

```
var kisi = ("Ahmet", "Aksoy")
```

```
var nokta = (x:10,y:20)
```


```
var ad = kisi.0 // Ahmet
```

```
var x = nokta.x // 10
```

```
var soyad = kisi.1 // Aksoy
```


```
var y = nokta.y // 20
```

## Veri Atama



```
kisi.0 = "Mehmet"  
kisi.1 = "İkinci"
```

```
print(kisi) // ("Mehmet", "İkinci")
```



```
nokta.x = 100  
nokta.y = 200
```

```
print(nokta) // (x: 100, y: 200)
```

Elementlere Sonrada İsim verilebilir.

```
var hataMesaji = (404, "Not Found")
```

```
var (kod, mesaj) = hataMesaji
```

```
print(kod)    // 404
```

```
print(mesaj)  // Not Found
```

## İç İçe Tuples

```
var ogrenci : (Int,(Bool,String)) = (1256, (true, "Ahmet"))
```

Diagram illustrating nested tuple access:

- Dış : 0** (Outer index 0) points to the first element of the outer tuple, `1256`.
- İç : 0** (Inner index 0) points to the first element of the inner tuple, `true`.
- İç : 1** (Inner index 1) points to the second element of the inner tuple, `"Ahmet"`.
- Dış : 1** (Outer index 1) points to the entire inner tuple `(true, "Ahmet")`.

```
var okulNo = ogrenci.0 //1256
var siniftaMi = ogrenci.1.0 //true
var isim = ogrenci.1.1 //Ahmet
```

Teşekkürler...



kasım-adalan



kasimadalan@gmail.com



kasimadalan