

# Kotlin OOP Eğitimi

## İleri Kotlin

Kasım ADALAN

Elektronik ve Haberleşme Mühendisi  
Freelance Software Developer

# Eğitim İçeriği

1. Exception Nedir ?
2. Do - Try – Catch Yapısı
3. Thread

# Exception Nedir ?

- Derleyici Hatası (Compiler Error) : Derleme öncesi yakalanan hatalar
  - Örn: karakter hataları, sentaks hatası, ...
- Hata (Exception): Çalışma anında (runtime) gerçekleşen hatalar
  - Örn: Sistem hataları, cihaz hataları, dosya bulunamadı, dizi indeksi aşıldı, ...

# Exception Hata Ayıklama

# try catch

- Derleme sırasında oluşabilecek hatalar için kullanılır.
- Genelde kotlin input – output işlemleri için kullanılır. Yani veri alışveriş işlemlerinde kullanılır.
- Kullanılacak yer mutlaka hata fırlatmalıdır.

```
try{
```

```
//Kontrol edilecek kodlama buraya yazılır.
```

```
}catch (e:Exception){
```


```
//Hata oluşunca burası çalışır.
```

```
}
```

## try catch bloğu

```
val x = 10
val y = 0

try{
    println("Sonuç :  $\${x/y}$ ")
    println("İşlem Tamam")
} catch (e:Exception){
    println("İkinci sayı sıfır olamaz")
}
```



# Farklı catch blokları ile hata yakalama

```
val x = 10
val y = 2

var dizi = Array<Int>(size: 2){0}

try{
    println("Sonuç : ${x/y}")
    println("İşlem Tamam")
    dizi[4] = 8
}catch (e:Exception){
    if ( e is ArithmeticException){
        println("İkinci sayı sıfır olamaz")
    }

    if ( e is ArrayIndexOutOfBoundsException){
        println("Dizinin boyutunu aştınız")
    }
}
```

```
val x = 10
val y = 5

var dizi = Array<Int>(size: 2){0}

try{
    println("Sonuç : ${x/y}")
    println("İşlem Tamam")
    dizi[1] = 8
}catch (e:ArithmeticException){
    println("İkinci sayı sıfır olamaz")
}

catch (e:ArrayIndexOutOfBoundsException){
    println("Dizinin boyutunu aştınız")
}
```

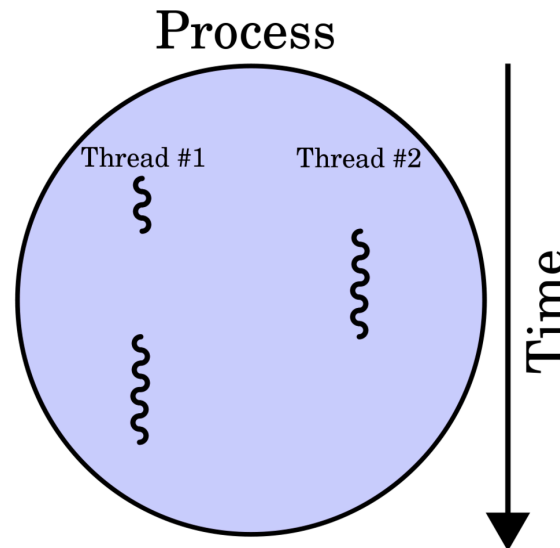
# Thread



# Thread

# Thread (Multi Tasking – Çok işlemcilik )

- Tek bir program akışı içerisinde birden fazla işlemin gerçekleştirilmesi Java'da Thread sınıfıyla mümkündür.



# Örnek Kullanımlar

- Web sunucular her istek geldiğinde yeni bir thread oluşturur.
- Microsoft Outlook gelen kutusunu kontrol ederken, yeni bir mesaj yazabilirsiniz.

# Sleeping - Uyutma

- Thread.sleep(int **milliseconds**)
- Milisaniyenin 1/1000'idir.
- Static bir metoddur , her yerde çağrılabilir.
- Yalnızca içinde bulunduğu thread'i etkiler.

## Thread Oluşturmak için İki Yöntem

- Thread sınıfından türetme:
  - **class BenimThreadim : Thread() {.....}**
  - `val is = BenimThreadim()`
  - `is.start()` ile başlar
  - Yalnızca tek bir sınıftan türetme yapılabilmektedir.
- Veya Runnable interface'ten türetme:
  - **class BenimThreadim : Runnable {.....}**
  - `val is = Thread(new BenimThreadim)`
  - `is.start()` başlar
  - `run()` metodunun override edilmesi gerekir.

# Örnek

```
class BirinciThread : Thread() {  
    public override fun run() {  
        for (i in 100..199){  
            println("Birinci Thread : $i")  
            Thread.sleep( millis: 100)  
        }  
    }  
}
```

```
class IkinciThread : Runnable {  
    public override fun run() {  
        for (i in 200..299){  
            println("Ikinci Thread : $i")  
            Thread.sleep( millis: 100)  
        }  
    }  
}
```

```
fun main(){  
    val birinciThread = BirinciThread()  
    birinciThread.start()  
    val ikinciThread = Thread(IkinciThread())  
    ikinciThread.start()  
    for (i in 100..199){  
        println("Main Thread : $i")  
        Thread.sleep( millis: 100)  
    }  
}
```

Teşekkürler...



kasım-adalan



kasimadalan@gmail.com



kasimadalan