eğitimden daha fazlası

wissen

# Swift & iOS
# Uygulama Geliştirme Eğitimi

# Enumerations

```
1. enum PusulaYönü {
2.     case Kuzey
3.     case Güney
4.     case Doğu
5.     case Batı
6. }
7.
```

# Enumerations

1. var istikamet = PusulaYönü.Kuzey
2. istikamet = .Güney

# Enumerations

```
1. switch istikamet {
2. case .Kuzey:
3.     println(”Kuzeye gidiyoruz!")
4. case .Güney:
5.     println(”Güneye gidiyoruz!")
6. case .Doğu:
7.     println(”Doğuya gidiyoruz!")
8. case .Batı:
9.     println(”Batıya gidiyoruz!")
10. }
```

# Enumerations

```
1. enum Barcode {
2.     case UPCA(Int, Int, Int, Int)
3.     case QRCode(String)
4. }
```

# Enumerations

1. var ürünBarkodu = Barcode.UPCA(8, 8509, 126, 3)
2. var biletBarkodu = .QRCode(”Davetiye No: 1234")

# Enumerations

```
1. switch ürünBarkodu {
2. case let .UPCA(numberSystem, manufacturer, product, check):
3.     println("UPC-A: \(numberSystem), \(manufacturer).")
4. case .QRCode(let productCode):
5.     println("QR code: \(productCode).")
6. }
```

# Enum Raw Values

- enum ASCIIControlCharacter: Character {
- case Tab = "\t"
- case LineFeed = "\n"
- case CarriageReturn = "\r"
- }

# Enumerations

```
1.  enum Gezegen: Int {
2.          case Merkür = 1
3.          case Venüs   // 2
4.          case Dünya   // 3
5.          case Mars    // 4
6.          case Jüpiter // 5
7.          case Satürn  // 6
8.          case Uranüs // 7
9.          case Neptün // 8
10. }
11. println(Gezegen.Neptün.rawValue)
12. // 8 yazdırır
```

# Enum Raw Values

- let dünyanınSırası = Gezegen.Dünya.rawValue
- // dünyanınSırası = 3

- let müstakbelGezegen = Gezegen(rawValue:7)
- // müstakbelGezegen'in tipi Gezegen?
- // ve değeri Gezegen.Uranüs

# Enum Raw Values

```
1.  let gezegenNo = 9
2.  if let birGezegen = Gezegen(rawValue:gezegenNo) {
3.      switch birGezegen {
4.      case .Dünya:
5.          println(”Dünyamız ☺")
6.      default:
7.          println(”Dünya dışında bi gezegen işte")
8.      }
9.  } else {
10.     println("\(gezegenNo) nolu bir gezegen yok!")
11. }
```

# Deneme Yapalım

# Protocol

1. protocol ÖrnekProtocol {
2. }

# Protocol

```
1. protocol ÖrnekProtocol {
2.     func kendiniTanıt()
3. }
```

# Protocol Kullanımı

```
1. class HarikaClass: ÖrnekProtocol {
2.     func kendiniTanıt() {
3.         println(“Ben harika bir CLASS’ım”)
4.     }
5. }
```

# Protocol Kullanımı

1. var a = HarikaClass ( )
2. a.kendiniTanıt()
3. // “Ben harika bir class’ım” yazdırır

# Protocol Kullanımı

```
1. struct HarikaStruct: ÖrnekProtocol {
2.     func kendiniTanıt() {
3.         println(“Ben harika bir STRUCT’ım”)
4.     }
5. }
```

# Using Protocol

1. var b = HarikaStruct ( )
2. b.kendiniTanıt()
3. // “Ben harika bir STRUCT’ım” yazdırır

# Class Only Protocols

```
protocol SomeClassOnlyProtocol: class {
    // İki noktadan sonra class yazınca, bu protocol
    // sadece class'lar tarafından kullanılabilir
}
```

# In-Out Parametreler

```
1. func ikiSayıyıDeğiştir(inout a: Int, inout b: Int) {
2.     let geçiciA = a
3.     a = b
4.     b = geçiciA
5. }
```

# In-Out Parametreler

1. var birinciInt = 3
2. var ikinciInt = 107
3. ikiSayıyıDeğiştir(&birinciInt, &ikinciInt)

4. println(birinciInt)  // 107 yazdırır
5. println(ikinciInt) // 3 yazdırır

# Generics

```
1. func ikiYazıyıDeğiştir(inout a: String, inout b: String) {
2.     let geçiciA = a
3.     a = b
4.     b = geçiciA
5. }
```

# Generics

```
1. func doubleDeğiştir(inout a: Double, inout b: Double) {
2.     let geçiciA = a
3.     a = b
4.     b = geçiciA
5. }
```

# Generics

```
1. func değerleriDeğiştir<T>(inout a: T, inout b: T) {
2.     var geçiciA = a
3.     a = b
4.     b = geçiciA
5. }
```

# Generics

```
1.  func tekrarla<Tip>(parça: Tip, sayı: Int) -> [Tip] {
2.      var sonuç = [Tip] ( )
3.      for i in 0..<sayı {
4.          sonuç.append(parça)
5.      }
6.      return sonuç
7.  }
8.  tekrarla(“Para”, 3)
9.  // [“Para”, “Para”, “Para”] şeklinde Array oluşur
10. tekrarla(10,2)
11. // [10, 10] şeklinde Array oluşur
```