AnkaSEC

Presented by :
Hatice Büşra Arslan

Java Syntax ve Değişkenler.

AnkaSEC

Sunumun Konuları

01 Java Syntax Kuralları

02 Değişkenler Ve Veri Tipleri
- Primitive Tipler (int, double, boolean, char)
- Referans Tipler (string)

03 Operatörler
- Aritmetik Operatörler
- İlişkisel Operatörler
- Mantıksal Operatörler

04 Tip Dönüşümleri (casting)

05 Scanner İle Konsoldan Veri Okuma

01: Java Syntax Kuralları

1. Java case-sensitive yani küçük/büyük harf duyarlılığına sahip bir dildir.
2. Her satır noktalı virgül (;) ile sonlandırılır.
3. Kod blokları süslü parantez “{ }” ile tanımlanır.
4. Programlar class içine yazılır.
5. Her açılan süslü parantez mutlaka kapanmalıdır.
6. Tek satırlı yorum için “//” kullanılır. Çok satırlı yorum için “/* yorum */” kullanılır.

```java
public class SyntaxOrnek {

    public static void main(String[] args) {
        int sayi = 5;

        System.out.println("Sayı değeri: " + sayi);

        if (sayi > 0) {
            System.out.println("Sayı pozitiftir.");
        }

        for (int i = 1; i<= 3; i++) {
            System.out.println("i = " + i);
        }
    }
}
```

02: Değişkenler ve Veri Tipleri

DEĞİŞKEN NEDİR?

- Bellekte veri tutmak için kullanılan isimlendirilmiş alanlara değişken denir.
- Java'da tip güvenliği (type safety) vardır.

```java
public class DegiskenOrnek {
    public static void main(String[] args) {

        int yas = 20;
        double notOrtalamasi = 3.45;
        boolean ogrenciMi = true;

        System.out.println("Yaş: " + yas);
        System.out.println("Not Ortalaması: " + notOrtalamasi);
        System.out.println("Öğrenci mi?: " + ogrenciMi);
    }
}
```

PRIMITIVE (İLKEL) VERİ TİPLERİ

- Primitive tipler doğrudan değeri tutarlar.
- Bazı çeşitleri şunlardır: "int, double, boolean, char"
- char tek tırnak (' ') kullanır. boolean sadece "true" veya "false" değerlerini alır.

```java
public class PrimitiveVeriTipleri {
    public static void main(String[] args) {

        byte byteDeger = 100; // byte → çok küçük tam sayılar (-128 / 127)
        System.out.println("byte: " + byteDeger);

        short shortDeger = 32000; // short → küçük tam sayılar
        System.out.println("short: " + shortDeger);

        int intDeger = 100000; // int → en çok kullanılan tam sayı tipi
        System.out.println("int: " + intDeger);

        long longDeger = 9_000_000_000L; // long → çok büyük tam sayılar (sonuna L konur)
        System.out.println("long: " + longDeger);

        float floatDeger = 3.14f; // float → ondalıklı sayı (sonuna f konur)
        System.out.println("float: " + floatDeger);

        double doubleDeger = 3.14159265359; // double → ondalıklı sayı (varsayılan)
        System.out.println("double: " + doubleDeger);

        char charDeger = 'A'; // char → tek karakter (tek tırnak)
        System.out.println("char: " + charDeger);

        boolean booleanDeger = true; // boolean → true / false
        System.out.println("boolean: " + booleanDeger);
    }
}
```

REFERANS TİPLERİ (STRING)

- Referans tipler nesnenin adresini tutar.
- string bir class'tır, çift tırnak " " kullanılır. Değiştirilemez (immutable)

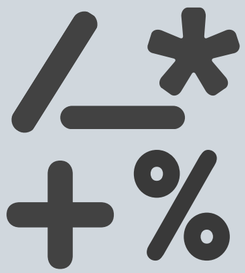
```java
public class StringOrnek {
    public static void main(String[] args) {

        String isim = "Ahmet";
        System.out.println(isim);

        isim = isim + " Yılmaz";
        System.out.println(isim);
    }
}
```

AnkaSEC

03: Operatörler



Aritmetik Operatörler

Aritmetik operatörler, programlama dillerinde sayısal değerler üzerinde matematiksel işlem yapmak için kullanılır.

- + iki sayısal değeri toplar. ++ değeri 1 arttırır.
- - bir değerden diğerini çıkarır. - - değeri 1 azalttırır.
- * iki sayıyı çarpar.
- / bir sayıyı diğerine böler.
- % bir sayının diğerine bölümünden kalanı verir.

İlişkisel (Karşılaştırma) Operatörler

İlişkisel (karşılaştırma) operatörler, iki değeri birbiriyle karşılaştırmak için kullanılır. Sonuç daima boolean'dır.

- == iki değerin birbirine eşit olup olmadığını kontrol eder.
- != iki değerin eşit olmadığını kontrol eder.
- > sol değerin sağ değerden büyük olup olmadığını kontrol eder.
- < sol değerin sağ değerden küçük olup olmadığını kontrol eder.
- >= sol değer büyükse veya eşitse true döner.
- <= sol değer küçükse veya eşitse true döner.

Mantıksal Operatörler

Mantıksal operatörler, birden fazla koşulu birlikte değerlendirmek, koşulların sonucunu true / false olarak birleştirmek veya tersine çevirmek için kullanılır.

- && VE operatörüdür. Her iki koşul da true ise sonuç true olur. Koşullardan biri bile false ise sonuç false olur.
- || VEYA operatörüdür. Koşullardan en az biri true ise sonuç true olur. Her ikisi de false ise sonuç false olur.
- ! DEĞİL operatörüdür. Koşulun sonucunu tersine çevirir.

03: Operatörler

Aritmetik Operatörler

- Örnek kod:

```java
public class AritmetikOperatorler {
    public static void main(String[] args) {
        int a = 10;
        int b = 3;

        System.out.println("Toplama (a + b): " + (a + b));
        System.out.println("Çıkarma (a - b): " + (a - b));
        System.out.println("Çarpma (a * b): " + (a * b));
        System.out.println("Bölme (a / b): " + (a / b));
        System.out.println("Mod (a % b): " + (a % b));
    }
}
```

- Örnek kodun ekran çıktısı:

```
Toplama (a + b): 13
Çıkarma (a - b): 7
Çarpma (a * b): 30
Bölme (a / b): 3
Mod (a % b): 1
```

İlişkisel (Karşılaştırma) Operatörler

- Örnek kod:

```java
public class IliskiselOperatorler {
    public static void main(String[] args) {
        int a = 10;
        int b = 5;

        System.out.println("a == b : " + (a == b));
        System.out.println("a != b : " + (a != b));
        System.out.println("a > b : " + (a > b));
        System.out.println("a < b : " + (a < b));
        System.out.println("a >= b : " + (a >= b));
        System.out.println("a <= b : " + (a <= b));
    }
}
```

- Örnek kodun ekran çıktısı:

```
a == b : false
a != b : true
a > b : true
a < b : false
a >= b : true
a <= b : false
```

Mantıksal Operatörler

- Örnek kod:

```java
public class MantiksalOperatorler {
    public static void main(String[] args) {

        boolean kosul1 = true;
        boolean kosul2 = false;

        System.out.println("kosul1 && kosul2 : " + (kosul1 && kosul2));
        System.out.println("kosul1 || kosul2 : " + (kosul1 || kosul2));
        System.out.println("!kosul1 : " + (!kosul1));
        System.out.println("!kosul2 : " + (!kosul2));
    }
}
```

- Örnek kodun ekran çıktısı:

```
kosul1 && kosul2 : false
kosul1 || kosul2 : true
!kosul1 : false
!kosul2 : true
```

04: Tip Dönüşümleri (Casting)

Implicit Casting

- Implicit Casting (Otomatik Tip Dönüşümü), Java'da daha küçük kapasiteli bir veri tipinin, daha büyük kapasiteli bir veri tipine Java tarafından otomatik olarak dönüştürülmesidir. Bu işlem güvenlidir ve veri kaybı olmaz.
- Küçük tip → Büyük tip

```java
public class ImplicitCastingOrnek {
    public static void main(String[] args) {

        int sayi = 10; // (küçük kapasite)
        double sonuc = sayi; // int → double (otomatik)

        System.out.println("int değer : " + sayi);
        System.out.println("double değer: " + sonuc);
    }
}
```

Explicit Casting

- Explicit Casting (Açık Tip Dönüşümü), Java'da daha büyük kapasiteli bir veri tipinin, daha küçük kapasiteli bir veri tipine programcı tarafından bilinçli olarak dönüştürülmesidir. Bu işlem manuel yapılır ve veri kaybı riski vardır.
- Büyük tip → Küçük tip

```java
public class ExplicitCastingOrnek {
    public static void main(String[] args) {

        double sayi = 10.75; // double (büyük kapasite)
        int sonuc = (int) sayi; // double → int (manuel)

        System.out.println("double değer: " + sayi);
        System.out.println("int değer : " + sonuc);
    }
}
```

Scanner ile Konsoldan Veri Okuma

Scanner Tanımlama

Scanner, Java'da kullanıcıdan klavye aracılığıyla veri almak için kullanılan hazır bir sınıftır.

- java.util paketi içinde yer alır.
- Konsol tabanlı uygulamalarda kullanılır.
- Konsoldan girilen veriyi okur ve uygun tipe çevirir.
- Kullanıcı etkileşimli (interactive) programların temelidir.
- "string, int, double, boolean" veri tiplerini okuyabilir.

Adım 1: Scanner Sınıfını Import Etmek

- Dosyada class tanımından önce scanner import edilir.

Adım 2: Scanner Nesnesi Oluşturmak

- Dosyada veri okunmadan önce main fonksiyonu içerisinde oluşturulur.

Adım 3: Kullanıcıdan Veri Okumak

- nextLine() → metin okur
- nextInt() → tam sayı okur
- nextDouble() → ondalıklı sayı okur

Adım 4: Alınan Veriyi Kullanmak

- Okunan veriler değişkenlerde saklanır.
- Programın geri kalanında kullanılır.
- Çıktı üretmek için System.out.println() ile yazdırılır.

```java
import java.util.Scanner;

public class ScannerOrnek {
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Adınızı giriniz: ");
        String ad = scanner.nextLine();

        System.out.print("Yaşınızı giriniz: ");
        int yas = scanner.nextInt();

        System.out.print("Boyunuzu giriniz (örn: 1.75): ");
        double boy = scanner.nextDouble();

        System.out.println("\n--- Kullanıcı Bilgileri ---");
        System.out.println("Ad: " + ad);
        System.out.println("Yaş: " + yas);
        System.out.println("Boy: " + boy);
    }
}
```

AnkaSEC

Presented by :
Hatice Büşra Arslan

Teşekkür Ederim!

arslanhbusra@gmail.com
https://l24.im/zpdt
hbusraarslan