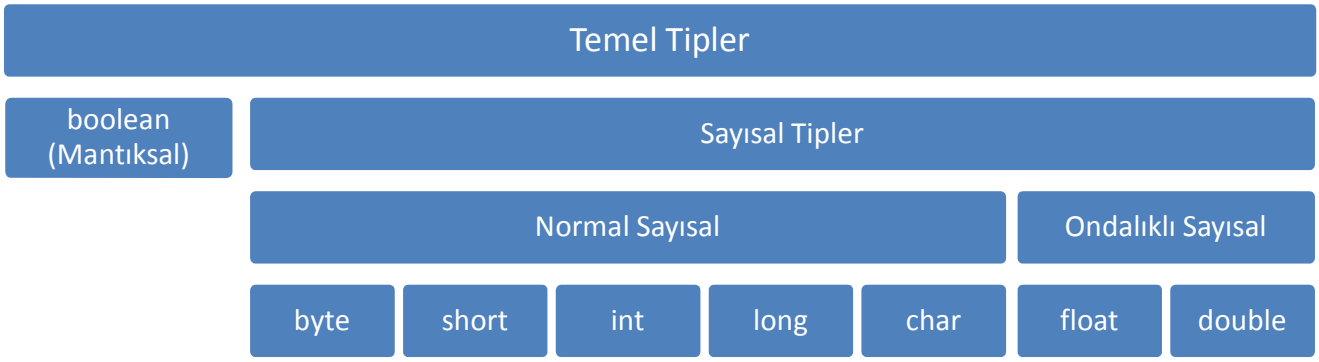
Veri Tipleri ve Değişkenler

Java içerisinde temelinde iki tip değişken vardır diyebiliriz. Bunlar primitive yani ilkel ve object yani nesne tipleridir diyebiliriz. Bunlar davranışsal ve bellekte saklandıkları alan tipleri olarak birbirlerinden farklıdır. İlkel tipler stack alanlarda tutulurken nesne tipleri heap alanlar içerisinde yer alır. Bu önemli bir noktadır stack alan üzerinde tutulan ilkel tipler direkt üzerlerinde değer taşırlar. Heap alanda tutulan nesneler tipleri ise bağlı referans adreslerini gösterirler.

Java içerisinde temel olarak 8 tipte ilkel tip vardır. Bunlar aşağıdaki gibidir.

Tip	Bellek Alanı (Bit)	Minumum Değer	Maksimum Değer	İlk Değer
byte	8	-128	+127	0
short	16	-32.768	+32.767	0
int	32	-2.147.483.648	+2.147.483.647	0
long	64	-9.223.372.036.854.775.808	+9.223.372.036.854.775.807	0L
float	32	32-bit IEEE 754	32-bit IEEE 754	0.0f
double	64	64-bit IEEE 754	64-bit IEEE 754	0.0d
char	16			
boolean	1	false	true	false

Yukarıdaki listede bulanan tiplerden byte, short, int ve long sayısal tipleri, float ve double ondalıklı sayısal tipleri, char karakteristik tipleri boolean ise mantıksal tipleri ifade eder.



Not : Boolean tipler 1 bitlik bir alana sahiptirler ve sadece true veya false değer alırlar üzerlerinde başka bir değer taşıyamazlar.

İlkel tipler direkt olarak üzerlerinde değer taşırlar. Bu yüzden hiç atama yapmazsanız sayısal değerlerde 0 mantıksal yani boolean değerde false olarak atama yapılır. Tabi local yani yerel değişkenler için geçerli değildir.

Bunlar dışında Java içerisinde ilkel tiplerin aşağıdaki gibi birer tane sarmallayıcı karşılıkları bulunmaktadır. Bu tipler referanslar üzerinden çalışır.

Tip	Sarmalayıcı (Wrapper) Tip
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Character
boolean	Boolean

Yukarıdaki tipte değişken tanımı yapılırsa ilkel tipten farklı olarak referans alanları üzerinden hareket edeler. Örnek olarak wrapper yani sarmallayıcı tipe atama yapmazsanız üzerindeki değer null olarak görünür. Null çıktısının anlamı herhangi bir referansa sahip olmadığıdır.

Not : Local değişkenler stack alandan farklı olarak local değişkenler için ayrılan bir stack alanda tutulurlar ve ilk değeri sizin atamanızı beklerler.

Dikkat : char karakteristik bir değişken olarak görünsede aslında sayısal bir değişkendir ve karakter tablosundaki ilgili karakter kodunu belirtir. Örneğin aşağıdaki kodun çıktısı karakter tablosundaki değerlerin toplami 65 + 66 = 131 yazdırır.

```
char x = 'A';
char y = 'B';
System.out.println(x + y);
```

Değişken tanımlama ve değer atama

Java uygulamalarında değişken tanımları aşağıdaki şekilde yapılmaktadır. Daha önceki konularda da belirttiğimiz gibi dinamik atama olmadığından dolayı değişken tipinide belirtmek zorundasınız.

[değişken tipi] [değişken adı] ;

Yukarıdaki standarta bir örnek verilecek olursa aşağıdaki gibi değişken tanımları çıkabilir.

```
int a;
float b;
Double c;
String d;
```

Yukarıdaki tanımlarda int ve float ilkel tipleri oluştururken Double ve String nesne tiplerini oluşturmaktadır. Aşağıdaki örnekde bu dört değişken sınıf içerisinde tanımlanıp çıktıları konsola bastırılıyor.

```
public class Ornek {
	int a;
	float b;
	Double c;
	String d;

	public static void main(String[] args) {
		Ornek orn = new Ornek();
		System.out.println(orn.a);
		System.out.println(orn.b);
		System.out.println(orn.c);
		System.out.println(orn.d);
	}
}
```

Bu örneğin çıktısı aşağıdaki gibidir.

```
0
0.0
null
null
```

Burada dikkat etmemiz gereken durum ilkel tipler 0 değerini alırken nesne tiplerinin null yazdırdığını yani üzerlerinde değer taşımadıklarını görüyoruz.

Not : Değişkenler tanımlanırken eğer aynı tipte birden fazla değişken tanımlanacaksa aşağıdaki gibi tanımlama yapılabilir.

```
int x, y, z;
```

Değer ataması

Değişken tanımlama sonrasında üzerlerine atanacak değerler = yani eşittir işareti ile belirtir az önceki örnekte tanımlanan değerlerin ataması aşağıdaki gibi yapılabilir.

```
int a = 2;
float b = 3.14F;
Double c = 22d;
String d = "Merhaba Dünya";
char e = 'A';
```

Eşitleme ile bir değer ataması yapılabileceği gibi başka bir değişkenin değeride bu atamaya dahil edilebilir. Örneğin a isimli bir değişkenin degeri b isimli bir değişkene aşağıdaki gibi atanabilir.

```
int a = 2;
int b = a;
```

Değişken belirtimlerinde eğer float, double ve long tanımlamalar yapıyorsak bunları özellikle belirtmeliyiz. Aslında tablo-1 üzerinde ilk değer atamalarında bunlar yer alıyordu ancak tekrar hatırlatmak gerekirse büyük küçük harf farketmeden float için F, double için D ve long için L tanımını belirtmemiz gerekir.

```
float x = 4.2F;
double y = 4.2D;
long z = 22L;
```

Dikkat : Herhangi bir tanım belirtimi yapmazsak sayısal değerler için int, ondalıklı değerler için double default olarak alınır.

Not : Java' değişken çevrimleri bir alt değerler için otomatik yapılır. Örnek olarak aşağıda long değer int değere atama yapabilirken int değer long değere atama yapamamaktadır.

```
int x = 4;
long y = x;
x = y ; //Bu satir hata verir
```

3. satırda hata vermesinin sebebi 32 bit lik int tipinde bir alana 64 bit lik long bir değerin atanmasına izin verilmemesidir. Aksi bir durumda yani long tipindeki alana int değer ataması yaparsak hata oluşmamaktadır.

Değişken geçerlilik alanları

Java içerisinde tanımlanan değişkenlerin geçerlilik alanları tanımlarının yapılması ile başlar ve scope yani kapsamları boyunca devam eder. Aşağıdaki örnekte x değişkeni bir kere tanımlandığı için tekrar tanımlandığında hata veriyor ancak y değişkeni tekrar tanımlandığında hata almıyoruz. Bunun nedeni y değişkeninin bulundugu kapsam süslü parantez ile bitiyor. Bundan sonra y değişkeninin geçerliliği kalmadığı için tanımlanmasında bir hata vermiyor.

```
public class Ornek {
	public static void main(String[] args) {
		int x = 5;
		{
			int y = 4;
			int x = 6; // Bu satır hata verir
		}
		int y = 8;
	}
}
```