

Java Platformunun Temel Özellikleri

Arakodlu Çalışma Sistemi: C ve C++ gibi programlama dillerinde programı derleyip çalışabilen dosya elde ettiğimizde bu dosyanın içerisinde microişlemcinin doğrudan çalıştırabileceği makine kodları bulunur. Halbuki Java programlama dili ile yazılmış bir programı derlediğimizde oluşan çalışabilen dosyanın içerisinde hiç bir mikroişlemcinin dili olmayan yapay bir arakod bulunur. Bu arakoda “byte code” denilmektedir. Bu arakod doğrudan çalışmaz. İsmine JVM (Java Virtual Machine) denilen program tarafından gerçek makine komutlarına dönüştürülerek çalışır. Arakod çalıştırılmak istendiğinde Java platformunun ismine JRE (Java Runtime Environment) denilen bir alt sistem devreye sokulur. Önce arakod arakod JRE tarafından gerçek makine komutlarına dönüştürülür ve çalıştırılır. Arakodun gerçek makine koduna dönüştürülmesi sürecine JIT derlemesi (Just In Time Compilation) denilmektedir. Şüphesiz bu süreçte ufakta olsa bir performans kaybı oluşmaktadır. Ama yüksek seviyeli uygulamalarda bu kayıp gözükmeyebilir.

Arakodlu çalışmanın ne avantajları vardır? Bu sayede programlar platform bağımsız bir biçimde yazılıp çalıştırılabilir. Bu kavrama WORA (Write Once Run Anywhere) denilmektedir. Derlenip oluşturulan byte kod JVM yüklü herhangi bir sistemde çalıştırılabilir. Byte code herhangi bir işletim sisteme bağımlı değildir.

Geniş Bir Sınıf Kütüphanesi: Java platformunun geniş bir sınıf kütüphanesi vardır. Veritabanı, web programlama, çeşitli GUI (Graphical User Interface) işlemleri için sınıflar vardır.

Hızlı Uygulama Geliştirme Ortamı: Java hızlı uygulama geliştirme ortamı rapid application development) sunar. Bunun için çeşitli görsel araçlardan faydalanılır. Öğrenilmesi nispeten daha kolaydır.

Güvenli Çalışma Ortamı: Java' da yazılmış programların sisteme zarar verme olasılıkları çok daha azdır. Yazılan program yüzünden makinada problem çıkma olasılığı (birtakım bilgilerin silinmesi vs.) çok daha azdır.

Programlama Dillerinin Tarihsel Gelişimi

Programlama dilleri birbirinden etkilenecek şekilde geliştirilmiştir. Bu etkilenme genel olarak bir dilden alıntı yaparak ona yeni özellikler ekleme şeklinde olmaktadır. Programlama dilleri bir ağaç olarak düşünülürse ağacın en tepesinde Fortran bulunmaktadır. (1954-1957). Fortran dünyanın ilk yüksek seviyeli programlama dilidir. Fortran'ı Algol ve Cobol dilleri izlemiştir. C programlama dili 1970 yılında Bell laboratuvarlarında Dennis Ritchie tarafından tasarlanmıştır. Java programlama dili Sun firmasında çalışan James Gosling tarafından tasarlanmıştır.

Programlama Dillerinin Sınıflandırılması

Programlama dillerini 3(üç) biçimde sınıflandırabiliriz:

1. Seviyelerine göre sınıflandırma
2. Kullanım alanlarına göre sınıflandırma
3. Programlama modeline göre sınıflandırma

Seviyelerine Göre Sınıflandırma

Seviye (level) bir programlama dilinin insan algısına yakınlığının bir ölçüsüdür. Yüksek seviyeli (high-level) diller hem dil özellikleri hem de uygulama geliştirme olarak kolay öğrenilebilen, insan algısına yakın dillerdir. Alçak seviyeli (low-level) diller makinenin doğal çalışmasına daha yakındır. Olabilecek en aşağı seviyeli dil saf makine dilidir.

Kullanım Alanlarına Göre Sınıflandırma

Genel Amaçlı diller, Bilimsel ve mühendislik diller, Oyun ve animasyon dilleri, Yapay zeka dilleri, Web dilleri, Sistem Programlama dilleri vs...

Programlama Modeline Göre Sınıflandırma

Bir programı yazarken kullanılan genel model (paradigm) önemlidir. Bazı dillerde sınıf yoktur. Program çeşitli alt programların (fonksiyon) birbirlerini çağırması yoluyla yazılır. Bu tekniğe prosedürel model denir. Bazı dillerde sınıflar vardır. Program sınıflar kullanılarak yazılmaktadır. Bu modele nesne yönelimli model (object oriented model) denilmektedir. Bazı dillerde programlar sanki matematiksel formül yazıyormuş gibi yazılmaktadır. Bu dillere fonksiyonel (functional) diller, programlama modeline ise fonksiyonel programlama modeli denilmektedir. Bazı diller birden fazla modeli destekler. Bunlara çok modelli (multiparadigm) diller denilmektedir.

Buna göre Java, genel amaçlı, yapay zeka, web programlama, bilimsel ve mühendislik alanlarında kullanılabilen nesne yönelimli, fonksiyonel özellikleri olan (özellikle Java 8 ile birlikte) yüksek seviyeli bir programlama dilidir.

Anahtar Notlar: Hangi programlama dilinin yüzde kaç kullanıldığına ilişkin istatistikleri yapan Tiobe isimli bir şirket vardır. (<http://www.tiobe.com/index.php/content/paperinfo/index.html>). Daha geniş sınıflandırma için (http://en.wikipedia.org/wiki/List_of_programming_languages_by_type)

Temel Kavramlar

Çevirici Program, Derleyici ve Yorumlayıcı: Bir programlama dilinde yazılmış olan programı eşdeğer olarak başka bir dile dönüştüren programlara çevirici programlar (translator) denir. Çevirici programlarda kaynak dil (source language) ve hedef dil (target language) söz konusudur. Örneğin, Fortran programlama dilini C programlama diline dönüştüren program çevirici programdır. Burada kaynak dil Fortran, hedef dil C dir. Bir çevirici programda hedef dil alçak seviyeli bir dilse (saf makine dili, sembolik makine dili ya da arakod) bu tür çevirici programlara özel olarak derleyici (compiler) denilmektedir. JVM nin byte kodu makine koduna dönüştürme faaliyeti de aslında bir derlemedir. Bu işlem çalıştırma ortamı içerisinde bulunan JIT derleyici (Just In Time compiler) tarafından yapılmaktadır.

Bazı programlar hiç kod üretmeden doğrudan çalıştırılırlar. Bunları çalıştıran programlara yorumlayıcı (interpreter) denilmektedir. Bazı diller yalnızca derleyicilerle, bazı diller yalnızca yorumlayıcılarla, bazı diller de her ikisi ile birden çalışmaktadır. Java derleyici ile çalışan bir programlama dilidir. Bununla birlikte Java 11 ile birlikte Java yorumlayıcı ile de çalıştırılabilmektedir.

İşletim Sistemi: İşletim sistemi makinenin donanımını yöneten, bilgisayar ile kullanıcı arasında köprü

görevi gören bir sistem yazılımıdır. Bir işletim sisteminin bir çok görevi bulunmaktadır. Örneğin: programları çalıştırmak, dosya organizasyonu, çeşitli donanımsal aygıtları yönetmek vs. İşletim sistemlerini günümüzde masaüstü ve mobil olmak üzere iki gruba ayırabiliriz. Popüler masaüstü ve mobil işletim sistemleri ve firmaları şunlardır:

Firma	Masaüstü Sistem	Mobil Sistem
Microsoft	Windows	-
Apple	Mac OS X	IOS
Google	-	Android
Açık Kaynak Kodlu (open source)	Linux	Embedded Linux

Mobil Sistemlerde Kullanılan Programlama Dilleri ve Ortamları: Her mobil sistemin kendine özgü bir geliştirme dili ve ortamı vardır. Her ne kadar tek bir ortamda geliştirilen ve tümünde çalışabilen platformlar üzerinde çalışılsa da henüz daha tam bir başarı sağlanamamıştır.

Java programlama dili özellikle Android işletim sisteminde çalışacak uygulamaların geliştirilmesinde yaygın olarak kullanılmaktadır.

Açık Kaynak Yazılım Özgür Yazılım ve Mülkiyete Sahip Yazılım: Özgür yazılım (free software) ve açık kaynak (open source) yazılımın aralarında bazı farklar olmasına karşın temel prensipleri aynıdır. Bu prensipler:

- Bedavadır. Kullanabilmek için herhangi bir lisans ücreti ödenmesi gerekmez
- Kaynak kod sahiplenilemez, Kaynak kod üzerine ekleme yapıldığında kodlar kapatılamaz. Onların da açılması gerekir.
- Çoğaltılabilir, izin almadan kullanılabilir.

Bunun tam tersi mülkiyete sahip (proprietary) yazılımlardır. Bu yazılımlar para verilerek kullanılır. Çoğaltılamazlar.

IDE (Integrated Development Environment): Derleyiciler komut satırından çalıştırılan basit bir arayüze sahip programlardır. Aslında teorik olarak Java ile geliştirilecek her uygulama notepad gibi basit bir editörle yazılıp, komut satırından derlenebilir. Fakat pratikte bu şekilde program yazmak oldukça fazla zaman alır. Yazılım geliştirmeyi kolaylaştırmak için ismine IDE denilen özel yazılımlar bulunmaktadır. IDE'lerin içerisinde editörler, yardımcı araçlar, test araçları, çeşitli kodları otomatik olarak yazabilen araçlar vs bulunur. Java ile uygulama geliştirilirken kullanılabilecek çeşitli IDE ler bulunmaktadır. Bunlardan en yaygın olarak kullanılanları Eclipse ve IntelliJIdea IDE'leridir. Önceleri Netbeans isminde bir IDE de kullanılmaktaydı. Ancak Netbeans kullanımı oldukça azalmıştır.

Klavyede Kullanılan Karakterlerin İngilizce Karşılıkları

Sembol	İngilizce Karşılığı
+	plus
-	minus, hyphen, dash
*	asterisk
/	slash
\	back slash
.	period, dot
,	comma
:	colon
;	semicolon
“	double quote
'	single quote
(...)	parenthesis left, right, opening, closing
[...]	(square) parenthesis left, right, opening, closing
{...}	brace left, right, opening, closing
=	equal sign
&	ampersand
~	tilda
@	at
<...>	less than, greater than, angular bracket
^	caret
	pipe
_	underscore
?	question mark
#	sharp, number sign
%	percent sign
!	exclamation mark
\$	dollar sign
...	ellipsis

İşletim Sisteminin Java ile Uygulama Geliştirmeye Hazır Hale Getirilmesi

Aşağıdaki sıralamaya ve açıklamalara göre işlemler yapılırsa çok fazla detay gerektirmeden sistem Java ile uygulama geliştirmeye hazır hale getirilebilmektedir.

Java Geliştirme Ortamının Kurulması: Java geliştirme ortamı için birden fazla alternatif bulunmaktadır. Oracle firmasının ürünü olan JDK (son versiyonu JDK 16) istenilen işletim sistemi için indirilebilmektedir.

Ayrıca açık kaynak kodlu olarak ismine OpenJDK denilen ayrı bir ürün de bulunmaktadır.

Uygulama Geliştirirken Kullanılacak IDE lerin yapılandırılması

Bilindiği gibi Java ile uygulama geliştirmede pek çok IDE kullanılabilmektedir. Burada öncelikle yaygın olarak kullanılan Eclipse IDE programının kurulumu ve yapılandırılması anlatılacaktır. Kursun ilerleyen aşamalarında IntelliJ Idea ile geliştirme yapılacak, dolayısıyla onun da kurulumu ve yapılandırılması anlatılacaktır.

Eclipse IDE sinin kurulumu ve Yapılandırılması: Eclipse IDE si aslında genel amaçlı bir IDE olmasına karşın en çok Java ile geliştirilen uygulamalar için kullanılmaktadır. Ayrıca Eclipse IDE sinin MyEclipse isminde lisanslı versiyonu da bulunmaktadır. Eclipse IDE si çeşitli eklentilerle bir çok ortam için kullanılabilir duruma getirilebilmektedir. Eclipse IDE si işletim sistemine kurulum gerektirmeyen bir IDE dir. Çalıştırılabilir dosya ve diğer dosyalar da dahil tek bir klasör içerisinde bulunmaktadır. Eclipse IDE si nin çalışılan işletim istemi için bedava versiyonu www.eclipse.org sitesinden indirilebilir. JDK doğru şekilde kurulduktan sonra ayrıca bir yapılandırma gerektirmemektedir.

Dil Kavramı

Dil iletişimde kullanılan semboller topluluğudur. Bir dilin kurallarına gramer denir. Gramer kavramının çeşitli alt alanları vardır. Bunlardan en önemlileri sentaks (syntax) ve semantik (semantic) alanlarıdır. Bir olgunun dil olabilmesi için en azından sentaks ve semantiğe sahip olması gerekir.

Sentaks doğru yazıma ve doğru dizilime ilişkin kurallardır. Örneğin:

“I a student am”

cümlesinde sentaks hatası yapılmıştır. Örneğin:

```
System.out.println(“Merhaba Java”;
```

Java kodunda kapanan parantez konmadığından yazıma ilişkin hata yapılmıştır. Yani sentaks hatası yapılmıştır.

Dildeki doğru dizilmiş öğrelerin ne anlam ifade ettiğine ilişkin kurallara semantik denilmektedir.

Doğal diller ve programlama dillerinin her ikisinde de sentaks ve semantik kuralları vardır. Doğal diller de ayrıca morfoloji gibi bazı kavramlar da vardır.

Bilgisayar Dilleri ve Programlama Dilleri

Bilgisayar bilimlerinde kullanılmak üzere tasarlanmış dillere bilgisayar dilleri (computer languages) denir. Bir bilgisayar dilinde akış da varsa o zaman programlama dili (programming language) denilmektedir. Her programlama dili bir bilgisayar dilidir, ancak her bilgisayar dili bir programlama dili olmayabilir. Örneğin: HTML bir bilgisayar dilidir fakat akış olmadığı için programlama dili değildir. C, C++, Java, C# dillerinde akış olduğu için bu diller programlama dilleridir.

Merhaba Java Programı

İskelet java programı ile “Merhaba, Java” yazısını basan temel bir program şöyledir:

```
package csd;

class App {
    public static void main(String [] args)
    {
        System.out.println("Merhaba, Java");
    }
}
```

Programın Derlenmesi ve Çalıştırılması

Merhaba Java programı aşağıdaki adımlardan geçilerek derlenebilir:

1. Notepad gibi bir editörde yukarıdaki program yazılarak csd isimli bir dizin içerisinde App.java isimli bir dosya olarak kaydedilir. Java derleyicileri .java uzantılı dosyanın isminin .java dan önceki kısmı ile aynı isimli public bir sınıfın o dosyada bulunmasını zorunlu tutmaktadır. Bu sebeple yukarıdaki programın yazıldığı dosyanın ismi App.java olmalıdır. Ayrıca Java da bir sınıfı hangi paket içerisinde bildirilmişse üretilcek olan arakodun çalıştırılması için o paket isminde bir klasör içerisinde bulunması zorunludur. Bu konu ileride ele alınacaktır.

2. Komut satırına geçilir. Oradan programın bulunduğu dizine geçilir. (Programın bulunduğu dizin Merhaba Java uygulamasında csd dizinidir)

3. Komut satırından aşağıdaki komut uygulanır.

```
javac <.java dosyası>
```

Örnek:

```
javac App.java
```

“javac” Java programlama dili ile yazılmış programların derlenmesi için kullanılan derleyici programın ismidir.

Anahtar Notlar: Windows işletimde komut yorumlayıcı programın ismi cmd.exe programıdır. cmd.exe program için bazı komutlar şunlardır:

e: → Sürücü değiştirme komutu (Burada e sürüc harfi)

cd → Dizin değiştirme (Change Directory)

dir → İçerik görüntüleme

cd .. → Bir üst dizine git

Derlenmiş ve arakodu üretilmiş olan program aşağıdaki gibi çalıştırılabilir:

1. Arakodun bulunduğu dizinin bir üst dizinine geçilir. (Burada App.class arakodunun csd dizininde olması gerektiğinden csd dizininin bulunduğu dizine geçilmelidir)
2. Komut satırından aşağıdaki komut uygulanır:

java <dizin ismi (paket ismi)>.<class dosyasınının .class uzantısı hariç adı>

Örneğin:

java csd.App

“java” programı ile, javac ile üretilen byte kod çalıştırılabilmektedir.

Atom Kavramı

Bir programlama dilinde kendi başına anlamlı olan en küçük yapı birimine atom (token) denir. Atomlar 6(altı) gruba ayrılır:

1. Anahtar Sözcükler (Keywords / Reserved Words): Dil için özel anlmı olan ve değişken ismi olarak kullanılması yasak olan atomlardır. Örneğin: public, if, while int gibi

2. Değişkenler (Identifiers /Variables): İsmini programcının belli bir kurala belirlediği atomlardır. Örneğin: “Merhaba, Java” programındaki App ismi gibi.

3. Operatörler (Operators): Bir işleme yol açan ve işlem sonucunda bir değer üretilmesini sağlayan atomlardır. Örneğin: a * b ifadesindeki * operatör atomdur.

4. Sabitler (Literals / Constants): Program kodu doğrudan yazılmış sayılara denir. Örneğin: a * 8 ifadesindeki 8 sabit atomudur.

5. Stringler (String literals): İki tırnak içerisindeki yazılara tırnaklarıyla birlikte string atomu denir.

6. Ayraçlar (Delimiters/ Punctuators): Yukarıdaki gruplar dışında kalan tüm atomlara ayraç atomlar denilmektedir.

Boşluk Karakterleri (white space): Klavyeden boşluk duygusu yaratmak için basılan karakterlere boşluk karakterleri (white spaces) denilmektedir. Boşluk karakterleri tipik olarak SPACE, TAB, ENTER karakterleridir.

Java Programlama Dilinin Yazım Kuralı

Java programlama dilinin yazım kuralı iki madde ile özetlenebilir:

1. Atomlar arasında istenildiği kadar boşluk bırakılabilir.
2. Anahtar sözcükler ve değişkenler dışında kalan atomlar istenildiği kadar bitişik yazılabilirler (Anahtar sözcüklerle değişken arasında en az bir boşluk karakteri olmalıdır).

Derleyicilerin Hata Mesajları

Derleyicilerin hata mesajları 3(üç) gruba ayrılmaktadır:

1. Uyarılar (Warnings): Byte kodun (.class uzantılı dosya) oluşmasına engel olmayan, olası mantık hatalarının programcıya bildirilmesi amacıyla kullanılan mesajlardır. Programcı uyarıları dikkate almalıdır. Bir programın özel bir durum yoksa hiç uyarı mesajı alınmadan derlenmesi hedeflenmelidir.

2. Gerçek Hatalar (Errors): Dilin sentaks ve semantik kurallarına uyulmamasından kaynaklanan hatalardır. Bir programda tek bir gerçek hata olsa bile byte kod üretilmez. Şüphesiz programcı tüm gerçek hataları düzelttiğinde programı derleyebilir ve ara kod üretilir.

3. Ölümcül Hatalar (Fatal Errors): Derleme işleminin bitirilmesini dahi engelleyecek ciddi hatalardır. Programcının kod olarak yapabileceği bir şey yoktur. Genellikle sistemdeki problemler yüzünden ortaya çıkmaktadır. Örneğin: diskte boş alan olmaması, işletim sisteminde bir problem, bellekte yer olmaması gibi.

Sentaks için Genel Biçimler

Programlama dillerinin standartlarında ya da spesifikasyonlarında iki anlamlılığa (ambiguity) izin vermeyen BNF (Backus-Naur Form) notasyonu ya da türevleri kullanılmaktadır. Fakat bu notasyon biraz karışıktır. Kurs içerisinde, açısız parantez, köşeli parantez tekniği kullanılacaktır. Açısız parantezler içerisinde yazılanlar mutlaka olması gereken öğeleri, köşeli parantez içerisinde yazılanlar ise isteğe bağlı öğeleri belirtecektir. Bunların dışındaki tüm atomlar aynı pozisyonda aynı biçimde bulundurulmak zorundadır. Ayrıca, “burada bir takım öğeler olabilir ancak bizi ilgilendirmiyor” anlamına gelen aşağıdaki gibi bir biçim de kullanılacaktır:

//...

Burada // atomu yorum satırı oluşturmak için kullanılmaktadır. Yorum satırı kavramı ileride açıklanacaktır.

Merhaba Java Programının Açıklaması

Genel olarak söylemek gerekirse bir Java programı paketlerden (package), paketler sınıflardan (class), sınıflar da metotlardan oluşur. Bir sınıfı bir paket içerisinde bildirmek için o sınıfın .java dosyasının en tepesinde aşağıdaki gibi bildirim yapmak gerekir:

```
package <paket ismi>;
```

Bununla birlikte bir sınıf bir paket içerisinde ise diskte o paket isminde bir dizinin (folder) içerisinde bulunması gerekmektedir. Derleme sonucunda üretilen arakodun da paket isminde bir dizin içerisinde olması zorunluluktur. Paket kavramı daha sonra detaylı olarak ele alınacaktır.

Sınıf bildiriminin genel biçimi şöyledir:

```
[erişim belirleyici] class <sınıf ismi> {  
    //...  
}
```


Burada görüldüğü gibi sınıf bildiriminin başındaki erişim belirleyici olmayabilir. Sınıf kavramı daha sonra detaylı olarak ele alınacaktır.

Metot bildiriminin genel biçimi şöyledir:

```
[erişim belirleyici] [static] <geri dönüş değerinin türü><metot ismi>([parametre bildirimi])  
{  
    //...  
}
```

Java programlama dilinde alt programlara metot denir. Erişim belirleyici şunlardan biri olabilir:

```
public  
protected  
private
```

Erişim belirleyiciyi hiç yazılmayabilir (no modifier). Erişim belirleyiciler detaylı olarak ileride ele alınacaktır. Bir metot ayrıca static anahtar sözcüğü ile bildirilirse static metot olur.

Java programları main isimli bir metottan çalışmaya başlar.

Bir metodun bildirilmesi (declaration) demek onun programcı tarafından yazılması demektir. Çağırılması (call) ise çalıştırılması demektir. Static bir metodu çağırmanın genel biçimi şöyledir:

```
[paket ismi][.][sınıf ismi][.]<metot ismi>([argüman listesi]);
```

Örneğin:

```
csd.Sample.foo(10);
```

Ekrana yazdırmak için kullanılan metodlar static değildir. System standart bir sınıftır ve java.lang paketi içerisinde bulunmaktadır. out PrintStream sınıfı türünden bir referanstır. println ise PrintStream sınıfının static olmayan bir metottur. println metodu kendisine gönderilen argümanı ekrana basar. Merhaba Java programında iki tırnak içerisindeki yazıyı ekrana basmaktadır. Ayrıca PrintStream sınıfının print isimli bir metodu daha vardır. Bu metot da ekrana yazmak için kullanılır ancak ekrana yazma işleminden sonra imleci (cursor) yazının sonunda bırakır, println ise aşağı satırın başında bırakır.

Anahtar Notlar: Standart paketler ve sınıfların hepsi java isimli paketin altında bildirilmiştir. Çok kullanılan sınıflar da java isimli paketin içerisindeki lang isimli pakette bildirilmiştir. java.lang paketi içerisindeki sınıflar çok fazla kullanıldığından herhangi bir işlem yapmadan (import bildirimi, nitelikli kullanım vs) içerisindeki sınıflar doğrudan kullanılabilir. Bu konu ileride daha detaylı olarak anlatılacaktır.

Anahtar Notlar: Referans ve static olmayan metot kavramları daha sonra ele alınacaktır.

Bir Java programı pek çok paketten, pek çok sınıftan ve pek çok metottan oluşabilir. main metodu herhangi bir paket içerisindeki herhangi bir sınıf içerisinde olabilir. main metodu public ve static olmak zorundadır.

Anahtar Notlar: Kurs içerisinde metod ismi olarak *foo*, *bar*, *tar*, *zar*, *baz* gibi isimler sıklıkla kullanılacaktır. Bu isimlerin hiçbir anlamı yoktur. Tamamen uydurma isimlerdir. Pek çok kaynakta (kitap, makale vs.) bu isimler kullanılmaktadır.

Eğer çağrılan metodun (callee) bildirimi çağıran metodun (caller) bildirimi ile aynı paket içerisindeyse (farklı sınıflarda olabilirler) paket ismi hiç yazılmayabilir (istenirse yazılabilir). Örneğin:

```
package csd;

class App {
    public static void main(String [] args)
    {
        Sample.foo();
    }
}

class Sample {
    public static void foo()
    {
        System.out.println("foo");
    }
}
```

Çağrılan metod ile çağıran metodun bildirimi aynı sınıf içerisindeyse sınıf ismi de yazılmayabilir. (istenirse yazılabilir). Örneğin:

```
package csd;

class App {
    public static void main(String [] args)
    {
        Sample.bar();
    }
}

class Sample {
    public static void bar()
    {
        System.out.println("bar");
        foo();
    }

    public static void foo()
    {
        System.out.println("foo");
    }
}
```

Yorum Satırları

Bir programın kaynak kodunda derleyici tarafından dikkate alınmayan kod bölümlerine yorum satırları (comment lines) denilmektedir. Yorum satırları program içerisinde çeşitli açıklamalar yapmak amacıyla kullanılabilir. Program kodu içerisinde gerektiğinden fazla yorum satırı kullanmak kodun okunabilirliğini bozabilmektedir. Buna dikkat edilmesi gerekir. Java programlama dilinde yorum satırları iki biçimde oluşturulabilir:

1. // atomu ile oluşturulan yorum satırları yalnız o satırın // atomundan sonraki kısmının derleyici tarafından yok sayılması amacıyla kullanılır.
2. /* ve */ atomları arasında kalan kod bölümleri derleyici tarafından dikkate alınmaz. Bu şekilde birden fazla satır için yorum satırları oluşturulabilir.

Örneğin:

```
package csd;

class App {
    public static void main(String [] args)
    {
        /*
            Bu yazı derleyici
            tarafından dikkate
            alınmayacak

            */

        System.out.println("Merhaba, Java");

        //Bu satır derleyici tarafından dikkate alınmayacak
    }
}
```