Eclipse

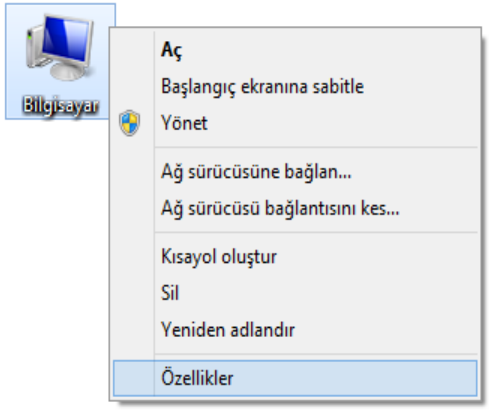
Bilgisayar Programlama

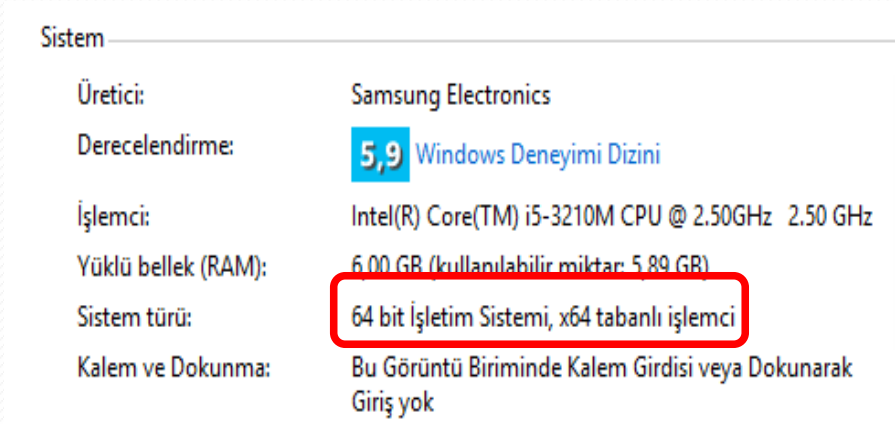
Eclipse nedir?

- Eclipse, açık kaynak kodlu bir yazılım geliştirme ortamıdır.
- Hızlı arayüzü, şık görünümü ve çok kuvvetli özellikleriyle kısa zamanda Java geliştiricileri arasında en popüler geliştirme ortamı olmuştur.
- En son kararlı versiyonu ECLIPSE IDE 2019-12'dir.

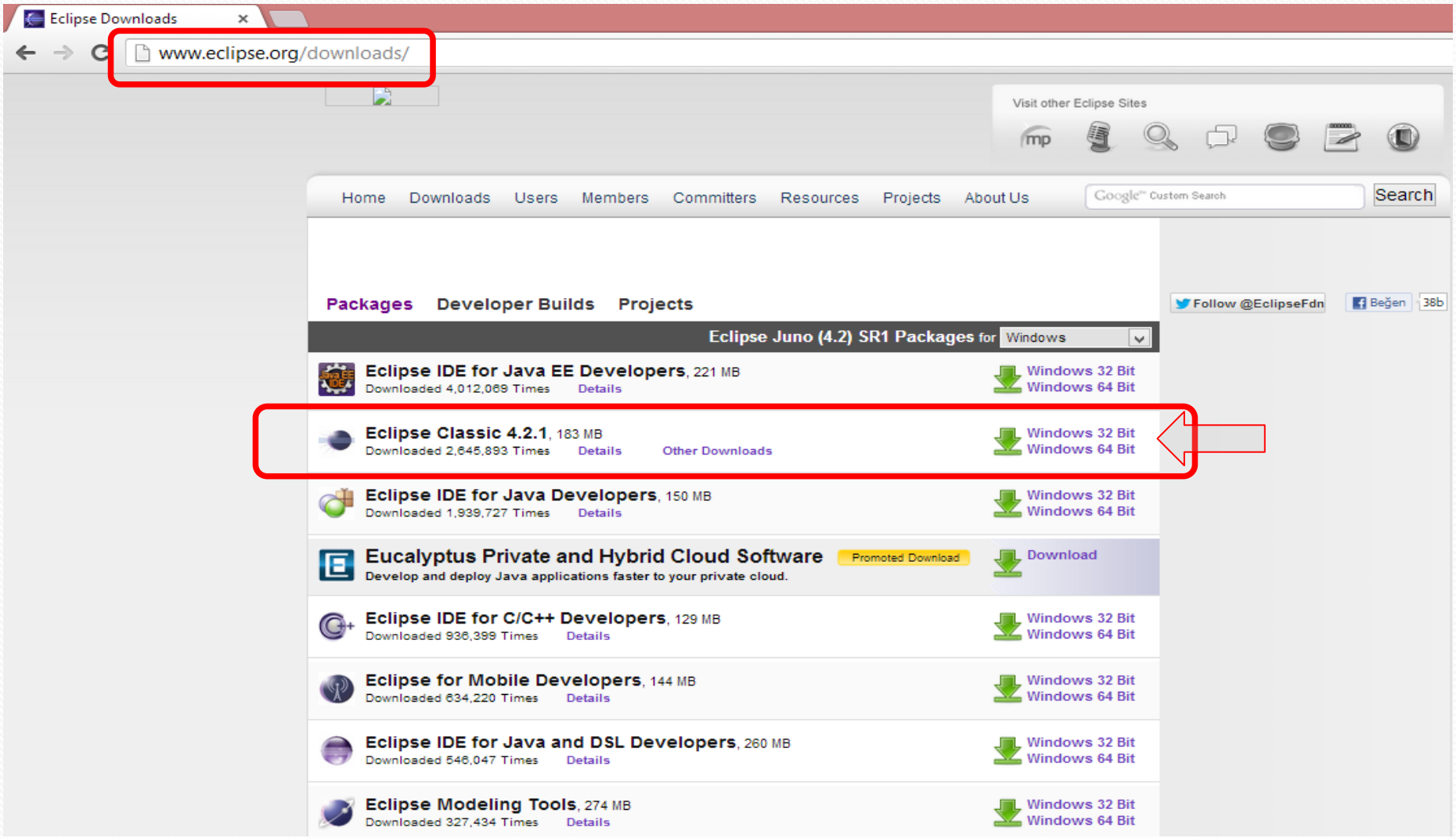
Eclipse Programını İndirmek

- http://www.eclipse.org/downloads/adresine girin.
- İşletim sisteminizin türüne (32 bit/ 64 bit) göre programı indirin.
- İşletim sisteminizin türüne "Bilgisayar"a sağ tıklayıp "Özellikler"den bakabilirsiniz.

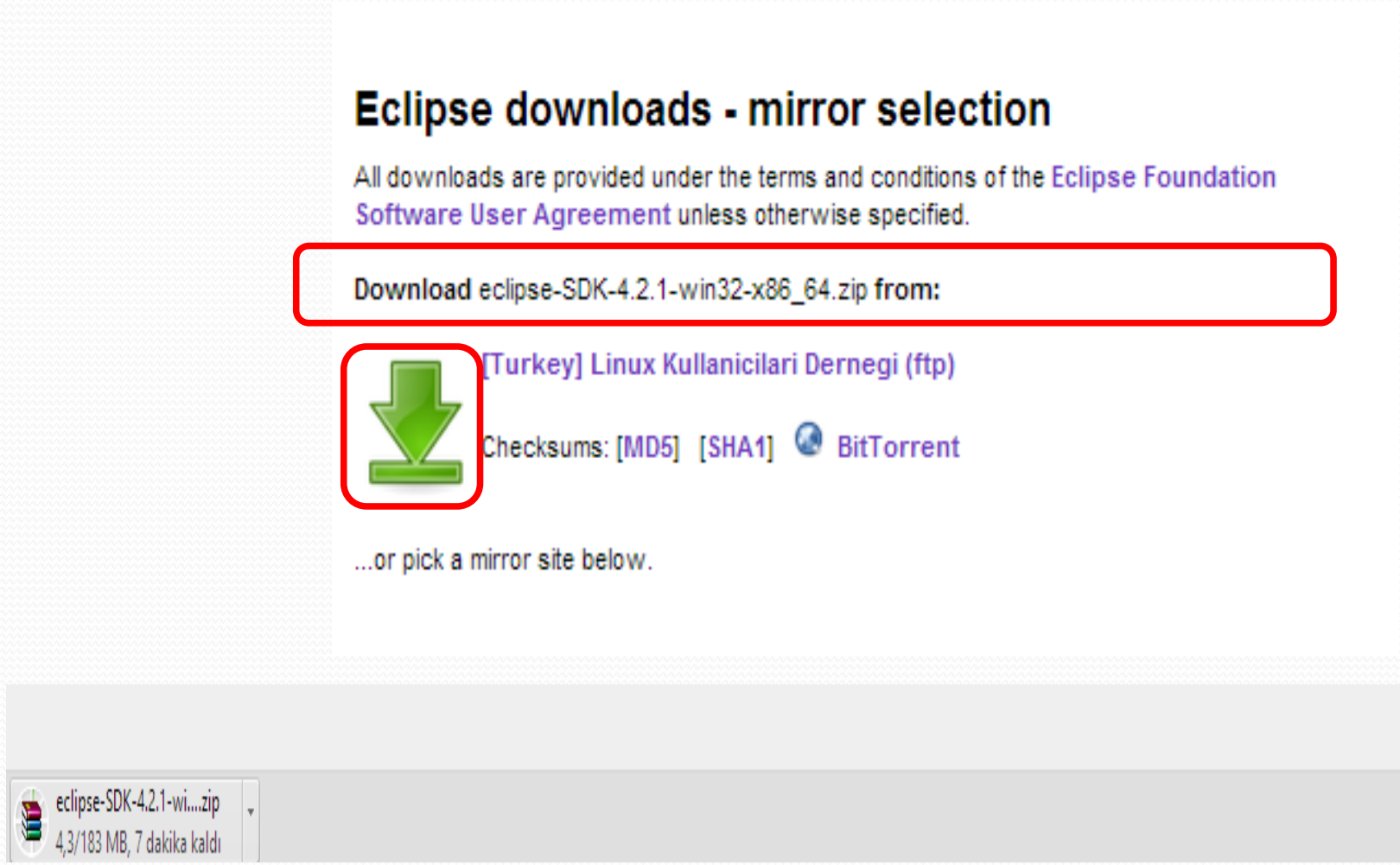




Eclipse Programını İndirmek

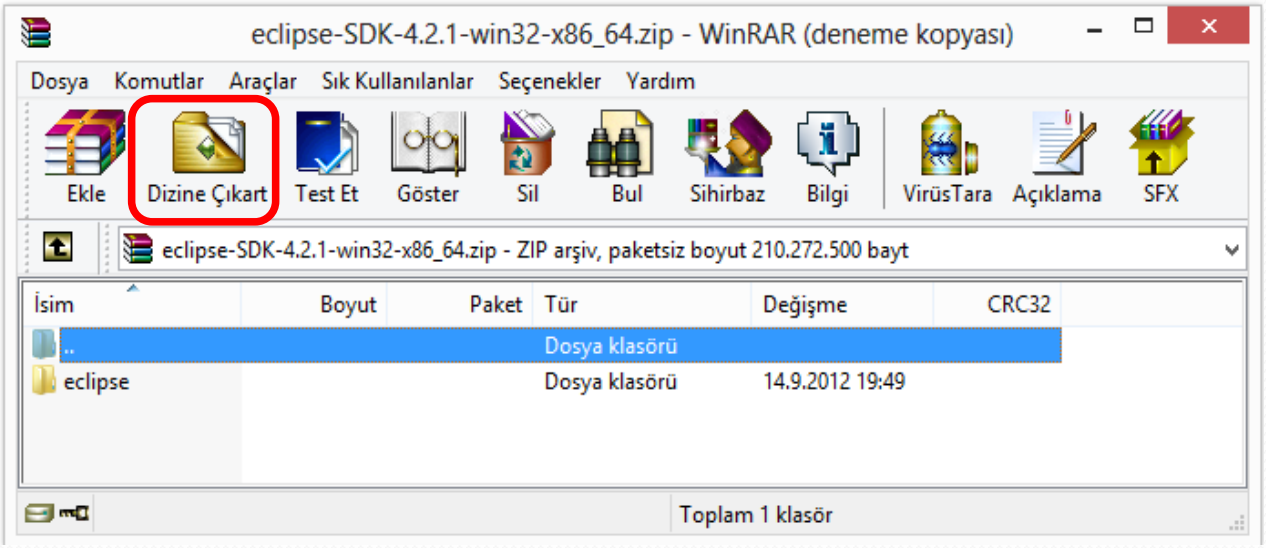


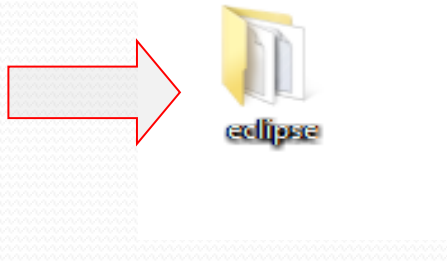
Eclipse Programını İndirmek



Eclipse Programını Kurmak

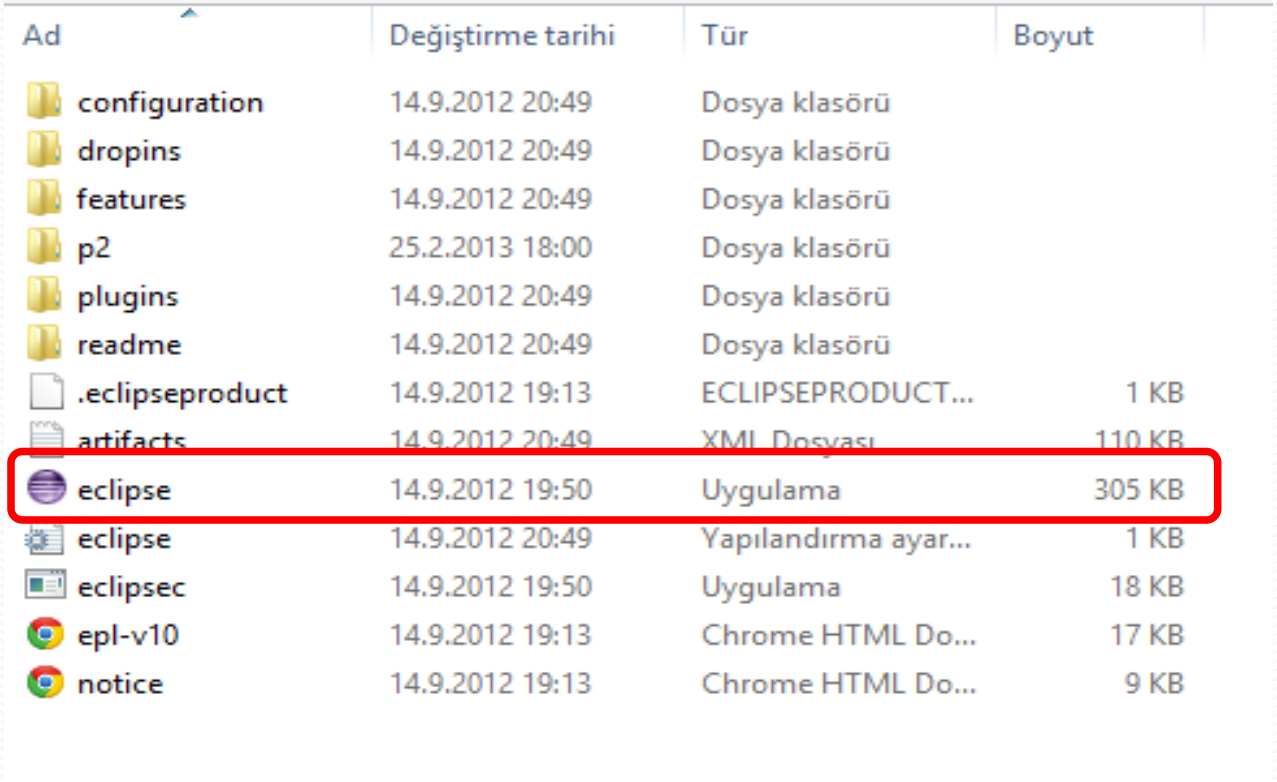
- Eclipse için kuruluma gerek yoktur. Sıkıştırılmış halde indirilen dosyaları açmak yeterlidir.





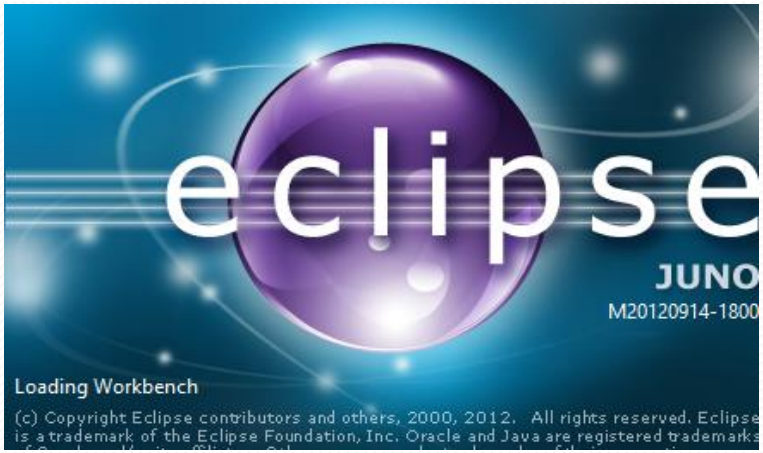
Eclipse Programını Açmak

- Çıkartılan klasörün içindeki dosyalar aşağıdaki gibidir. Uygulama dosyasına tıklanarak eclipse çalıştırılabilir.

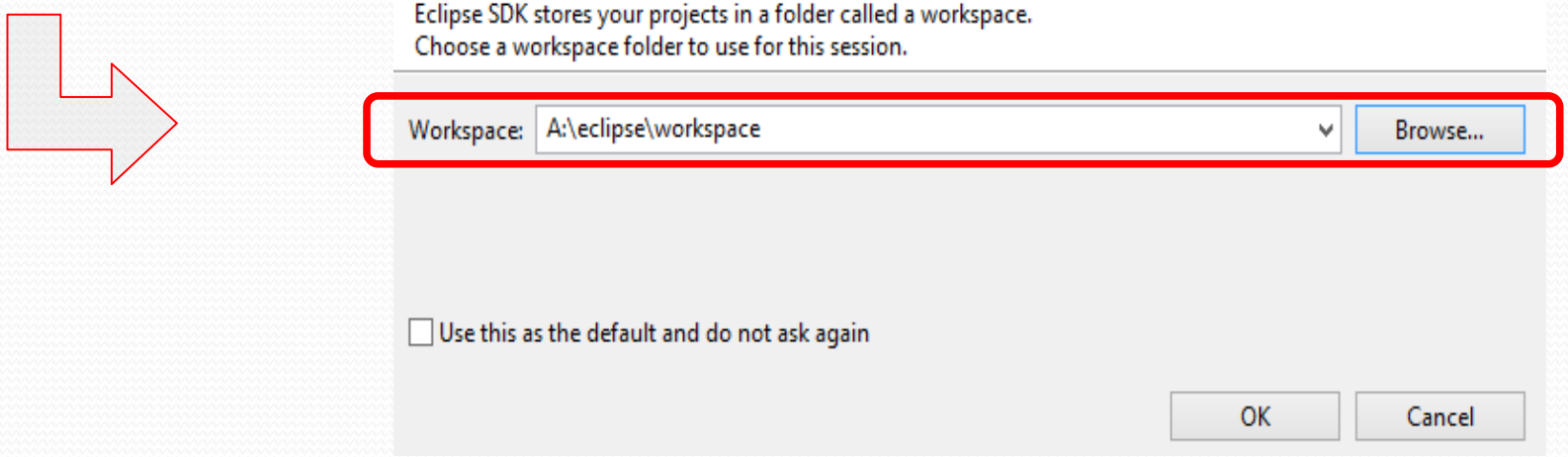


Ad	Değiştirme tarihi	Tür	Boyut
configuration	14.9.2012 20:49	Dosya klasörü	
dropins	14.9.2012 20:49	Dosya klasörü	
features	14.9.2012 20:49	Dosya klasörü	
p2	25.2.2013 18:00	Dosya klasörü	
plugins	14.9.2012 20:49	Dosya klasörü	
readme	14.9.2012 20:49	Dosya klasörü	
.eclipseproduct	14.9.2012 19:13	ECLIPSEPRODUCT...	1 KB
artifacts	14.9.2012 20:49	XML Dosyası	110 KB
eclipse	14.9.2012 19:50	Uygulama	305 KB
eclipse	14.9.2012 20:49	Yapılandırma ayar...	1 KB
eclipsec	14.9.2012 19:50	Uygulama	18 KB
epl-v10	14.9.2012 19:13	Chrome HTML Do...	17 KB
notice	14.9.2012 19:13	Chrome HTML Do...	9 KB

Eclipse Programını Açmak

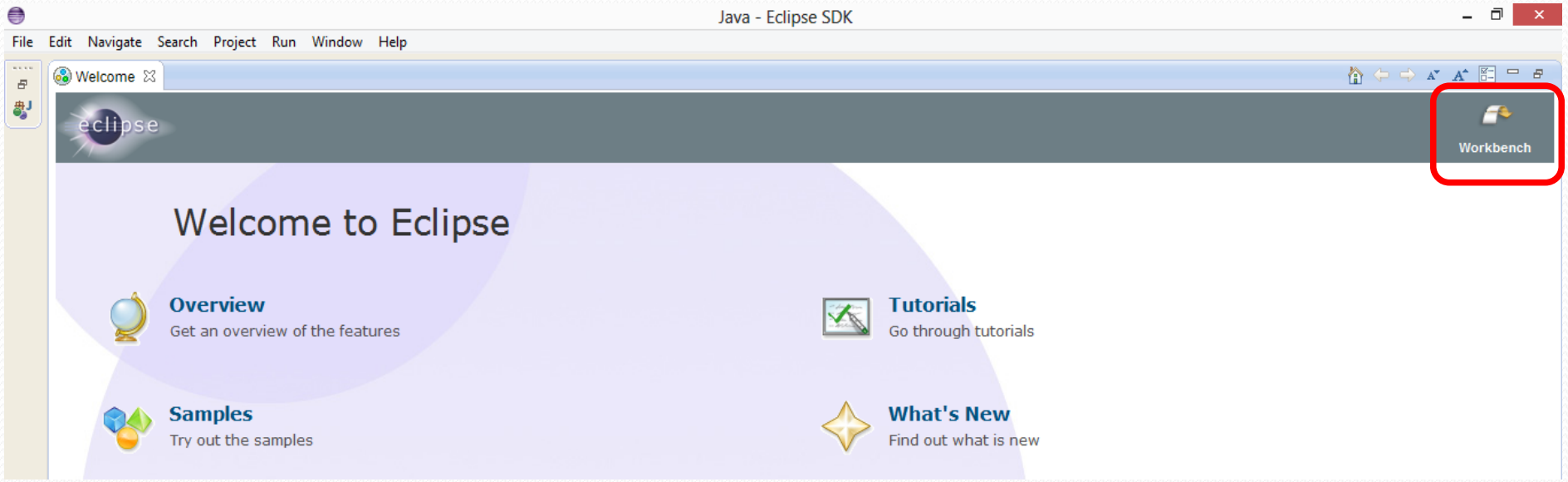


Eclipse tıklandıktan sonra açılan pencerede, **<u>çalışma dosyalarımızın kaydedileceği klasörü</u>** seçiyoruz. Bu klasör genellikle "workspace" şeklinde isimlendirilir.

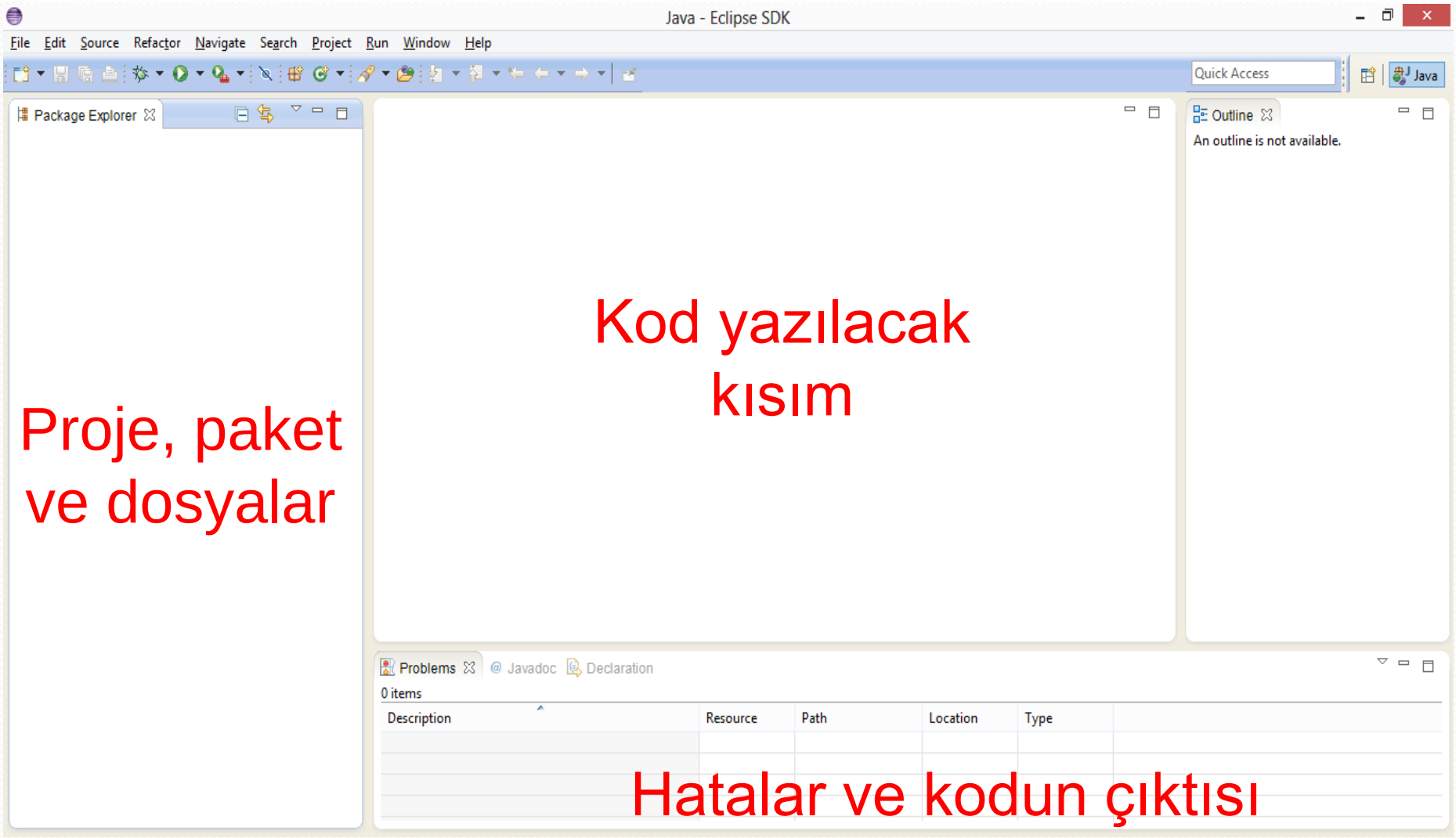


Eclipse Ortamı

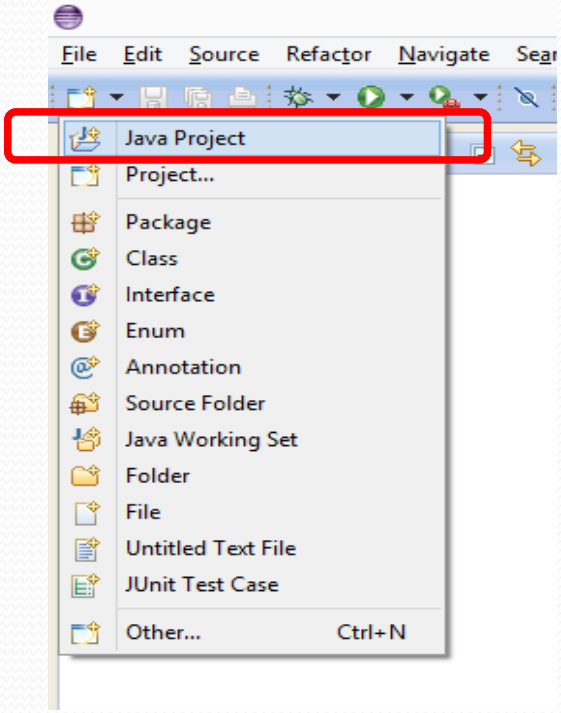
- Eclipse'in açılış ekranı aşağıda görüldüğü gibidir. Sağ üst taraftaki "Workbench" butonuna tıklayarak çalışmaya başlayabiliriz.

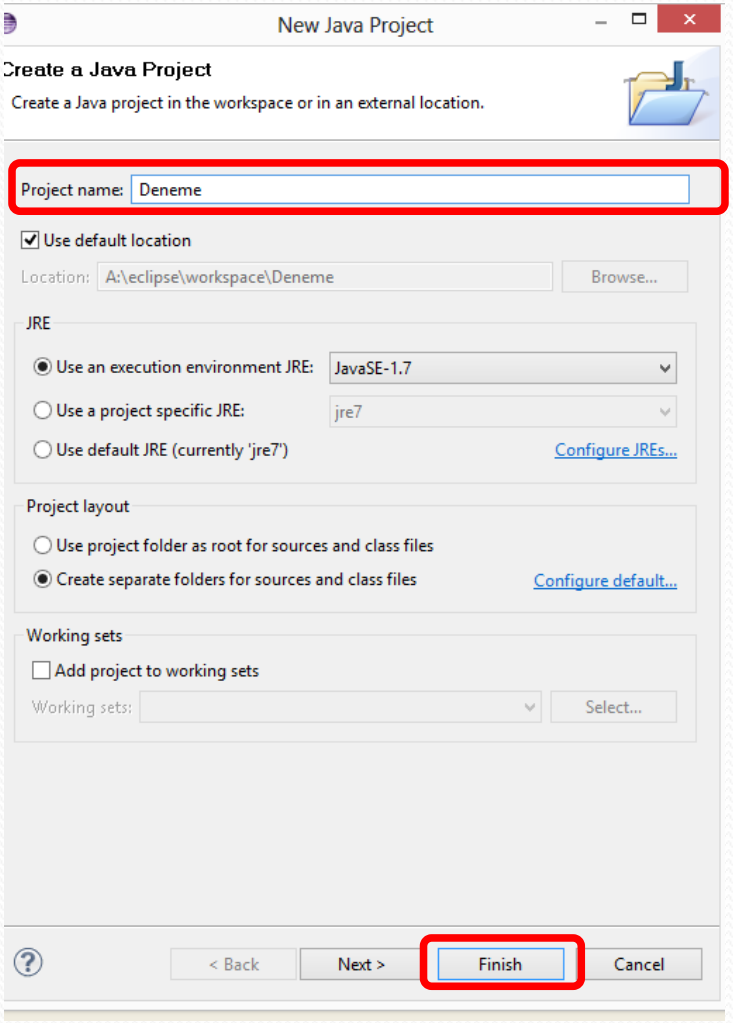


Eclipse Ortamı



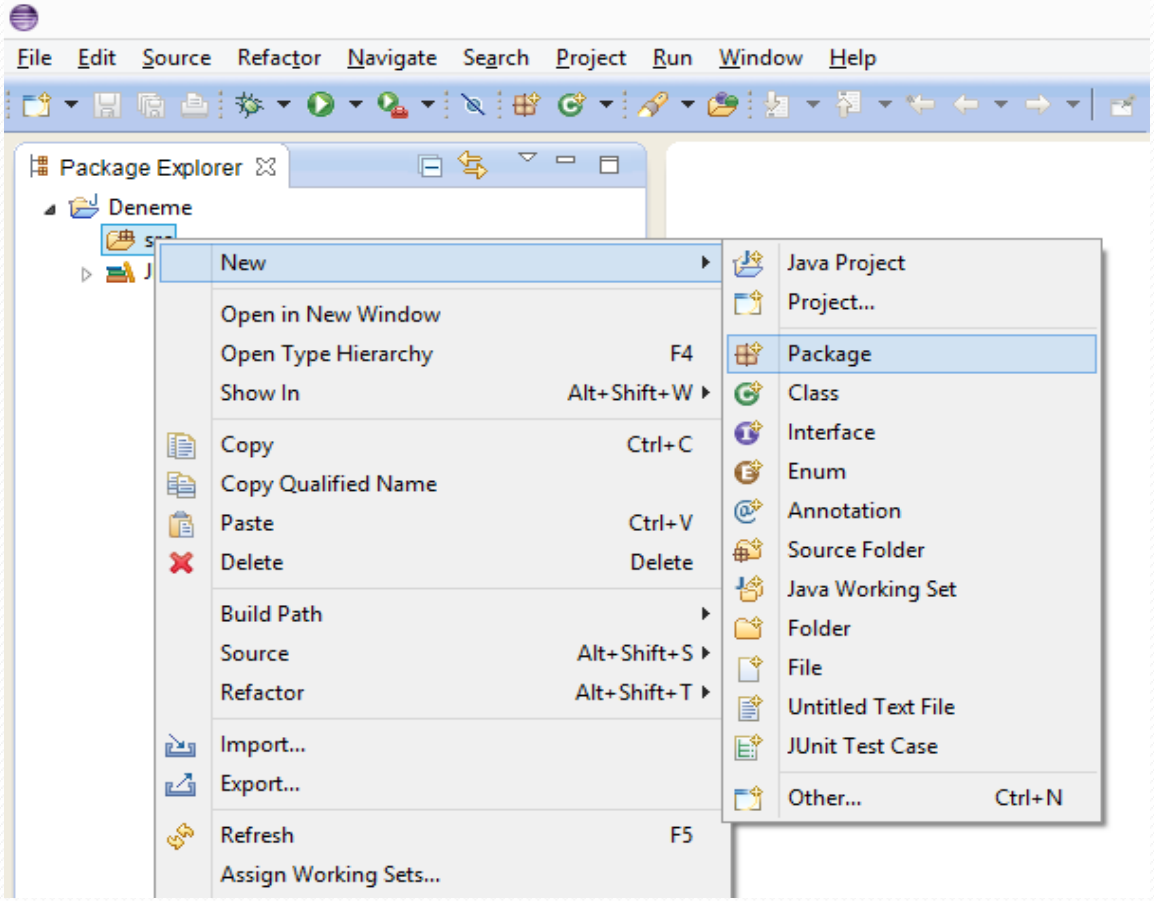
Proje Oluşturmak





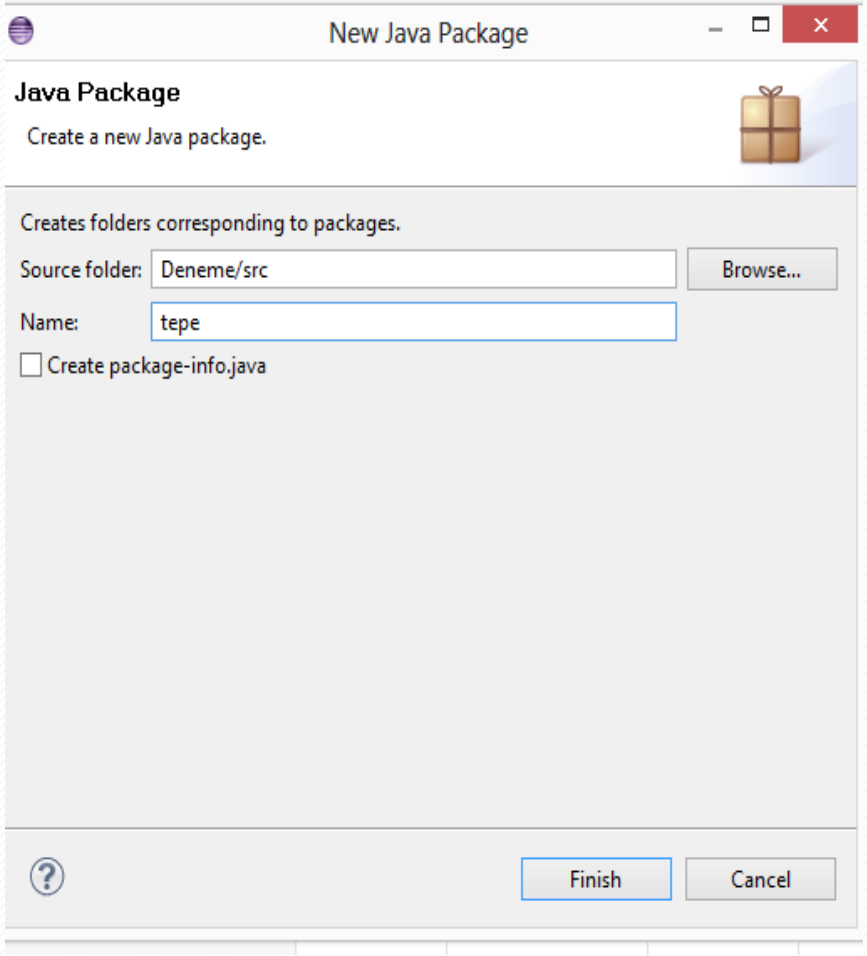
Paketler ve onların altında da dosyalar oluşturabilmek için öncelikle bir proje oluşturmamız gerekiyor.

Proje Oluşturmak

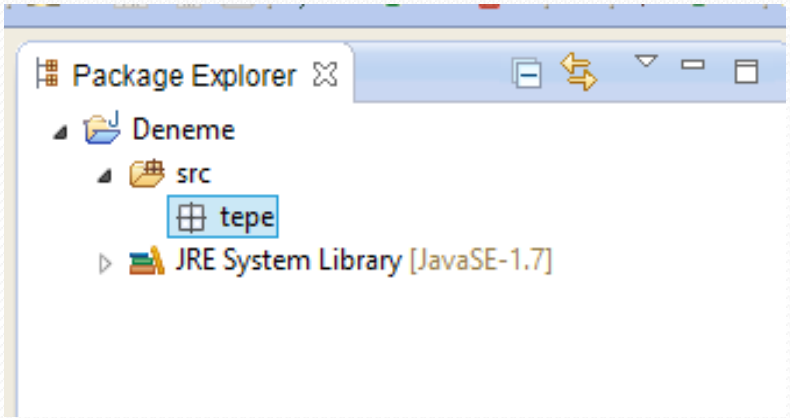


Proje oluşturduktan sonra altında yanda görüldüğü gibi **src (source)** isimli bir klasör belirecektir. Paketlerimizi ve java dosyalarımızı bu klasörün altında oluşturacağız.

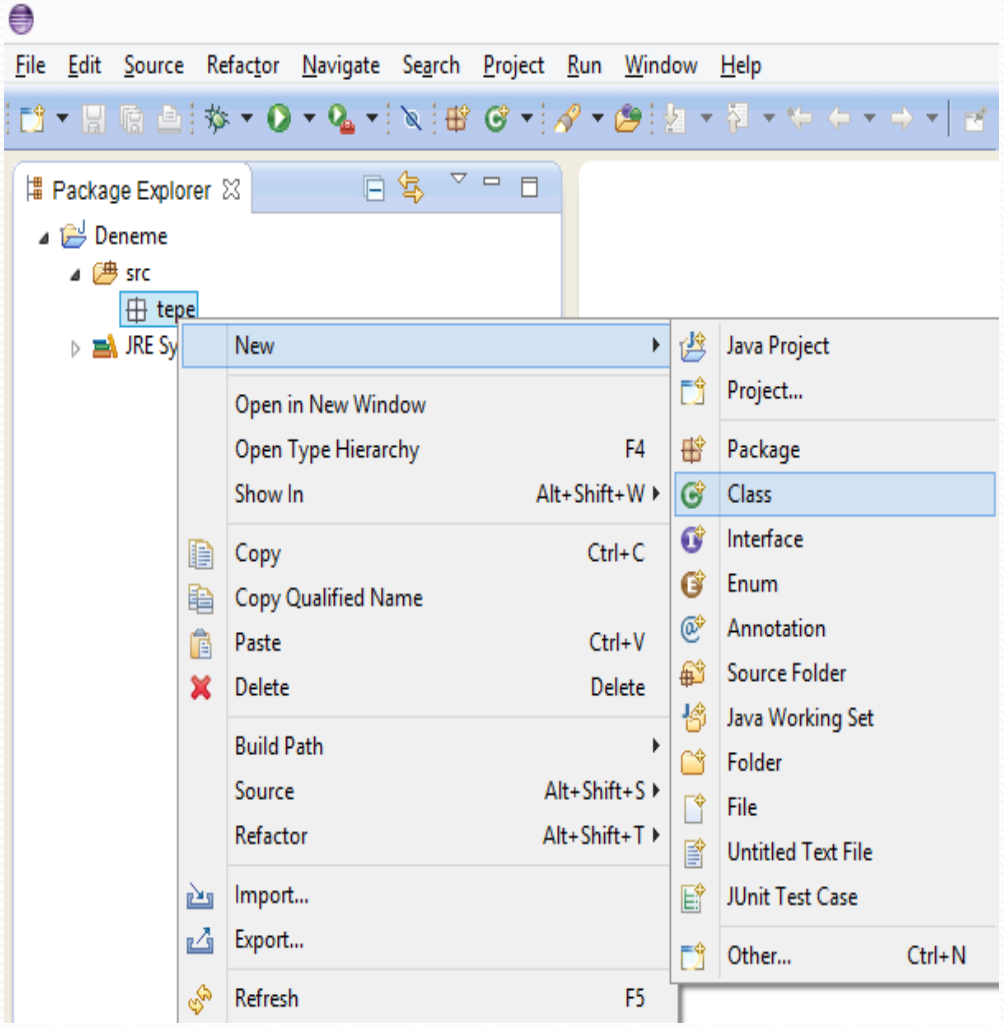
Proje Oluşturmak



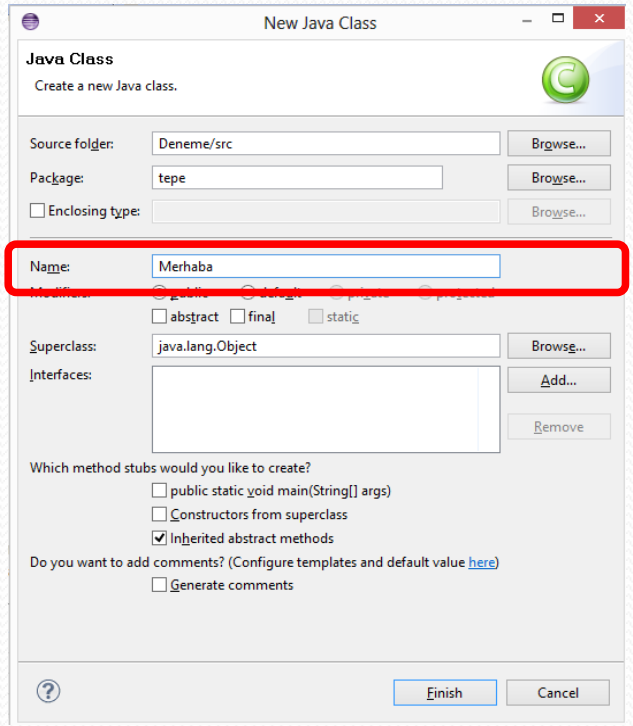
Paketimize isim verdikten sonra “Finish” butonuna tıklıyoruz ve aşağıda görüldüğü gibi paket ismi kaynak klasörün altında görünüyor.



Dosya Oluşturmak

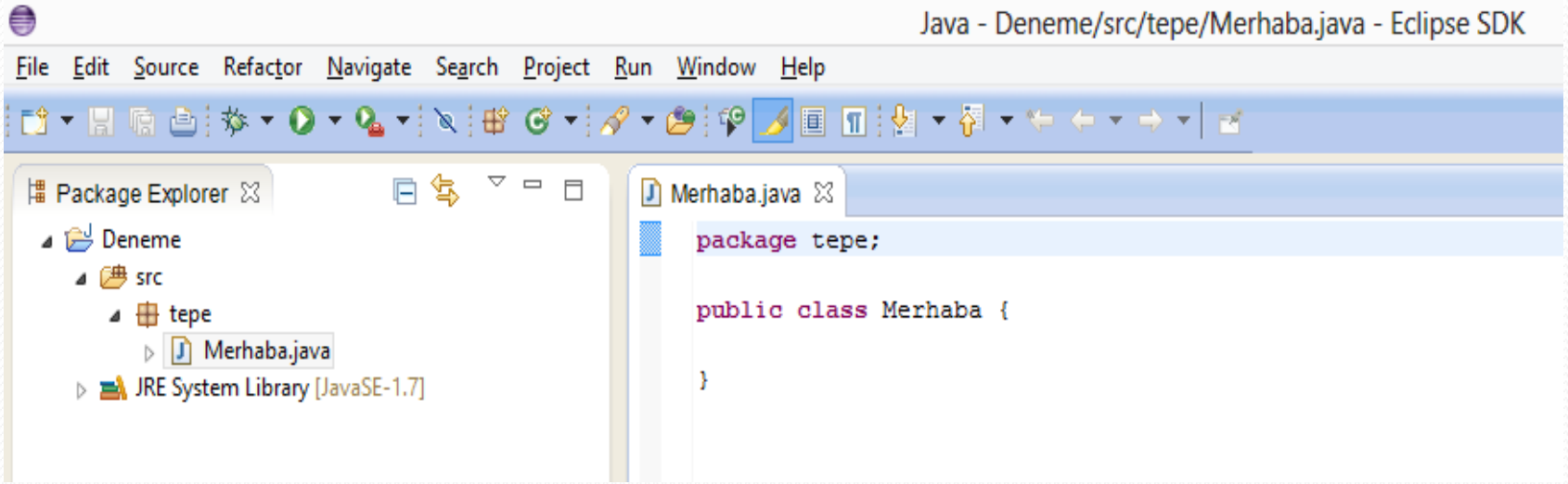


Paket isminin üzerine sağ tıklayarak yeni bir "Class" dosyası oluşturuyoruz ve dosyamıza isim veriyoruz.



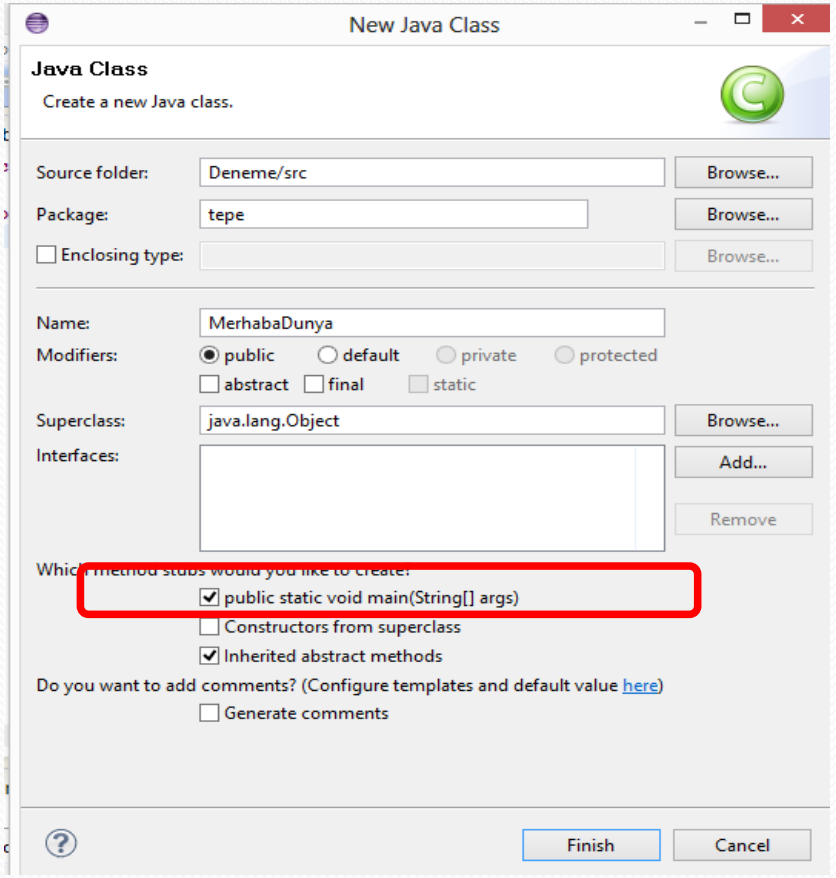
Kod Yazmak

- Yukarıda görüldüğü gibi artık paketimizin altında bir java dosyası görünüyor ve kod penceresinde de kaynak kodu görebiliyoruz. Artık bu orta pencere üzerinde kodumuzda istediğimiz değişiklikleri yapabiliriz.



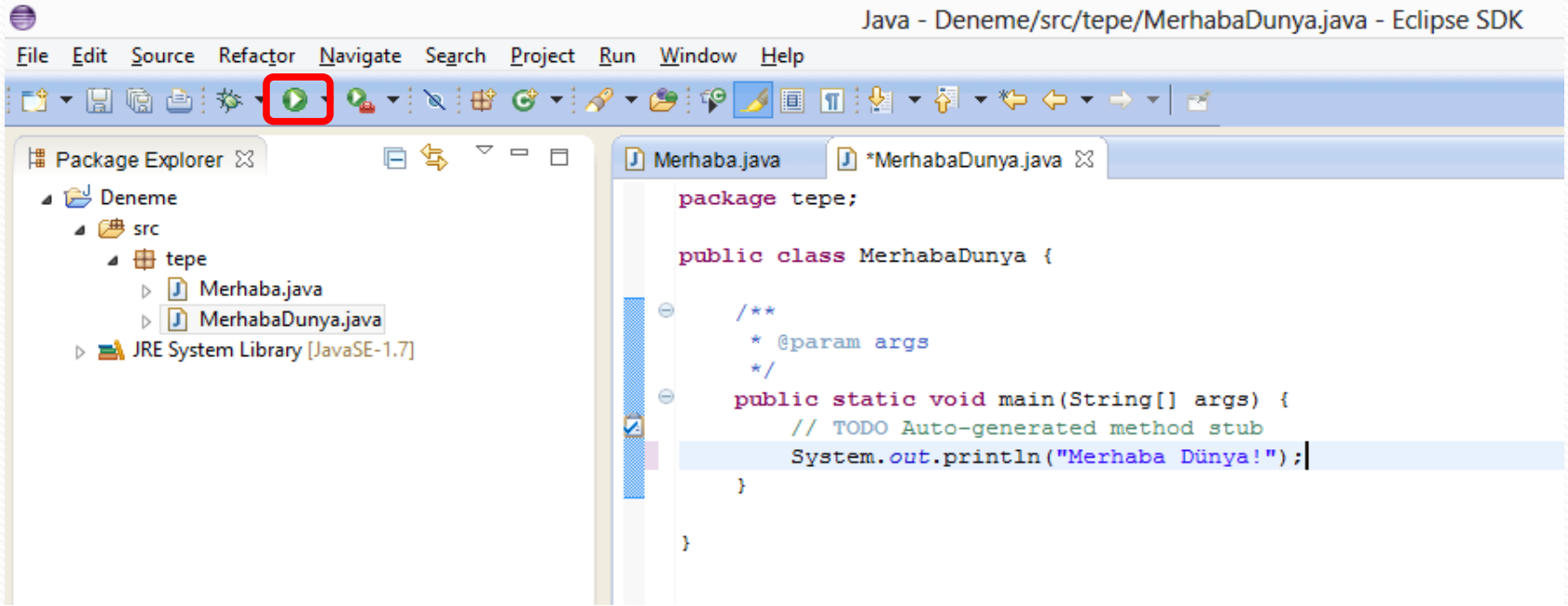
Kodu Çalıştırmak

- Bir önceki yansıdaki sınıf çalıştırılabilir değildi. Sınıfın çalıştırılabilir olması için oluşturma ekranında ilgili yeri tıklıyoruz ve main() bloğunun oluşmasını sağlıyoruz.



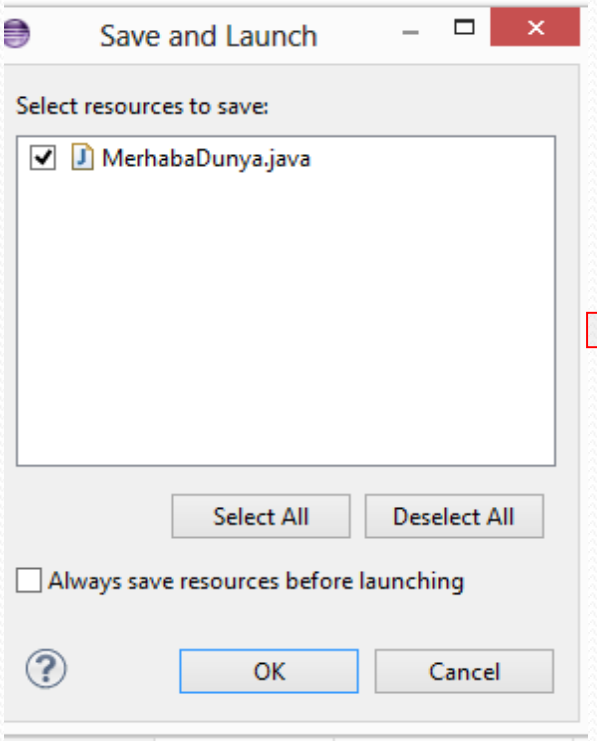
Kodu Çalıştırmak

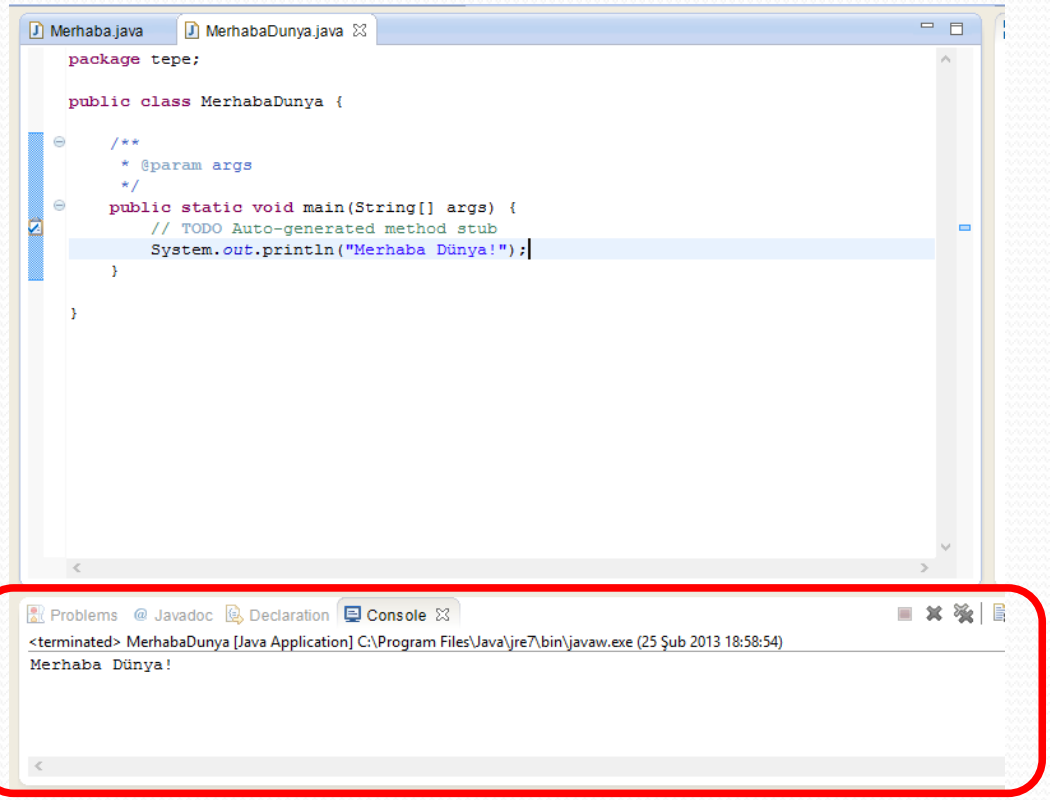
- Aşağıdaki kod çalıştırlabilir bir koddur; çünkü bir **main()** bloğuna sahiptir. Eclipse'de kodu çalıştırabilmek için araç çubuğundaki sağ ok simgesine tıklıyoruz **(Run)**.



Kodu Çalıştırmak

- Dosyayı çalıştırınca Eclipse dosyayı önce kaydetmemiz için uyarıyor. Kaydediyoruz ve çalıştırıyoruz.





Hataları Görmek

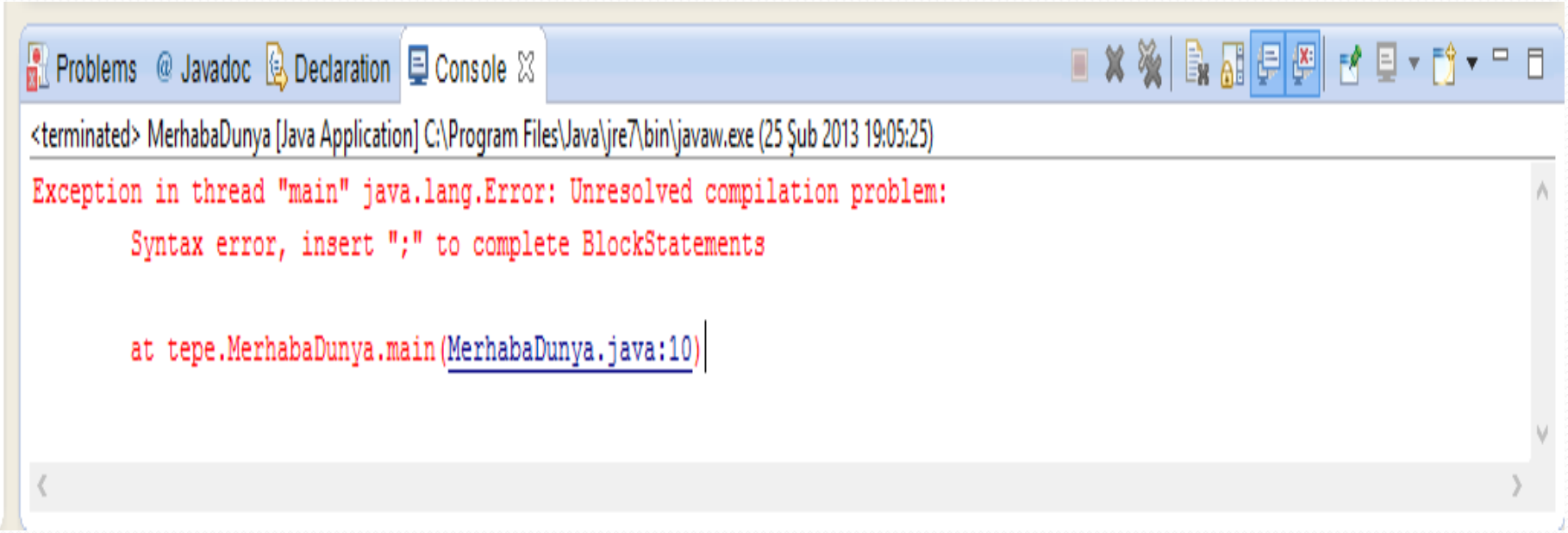
- Eclipse'in en önemli özelliklerinden birisi koddaki hatayı anında göstermesidir. Örneğin satır sonundaki noktalı virgülü unuttuğumuz zaman görüntü aşağıdaki gibi olur:

*MerhabaDunya.java

```
package tepe;

public class MerhabaDunya {

    public static void main(String[] args) {

        System.out.println("Merhaba Dünya!")

    }

}
```

Hataları Görmek

- Kodu bir önceki yansıda olduğu şekliyle çalıştırdığımızda aşağıdaki gibi hata alırız:



Temel Hatalar

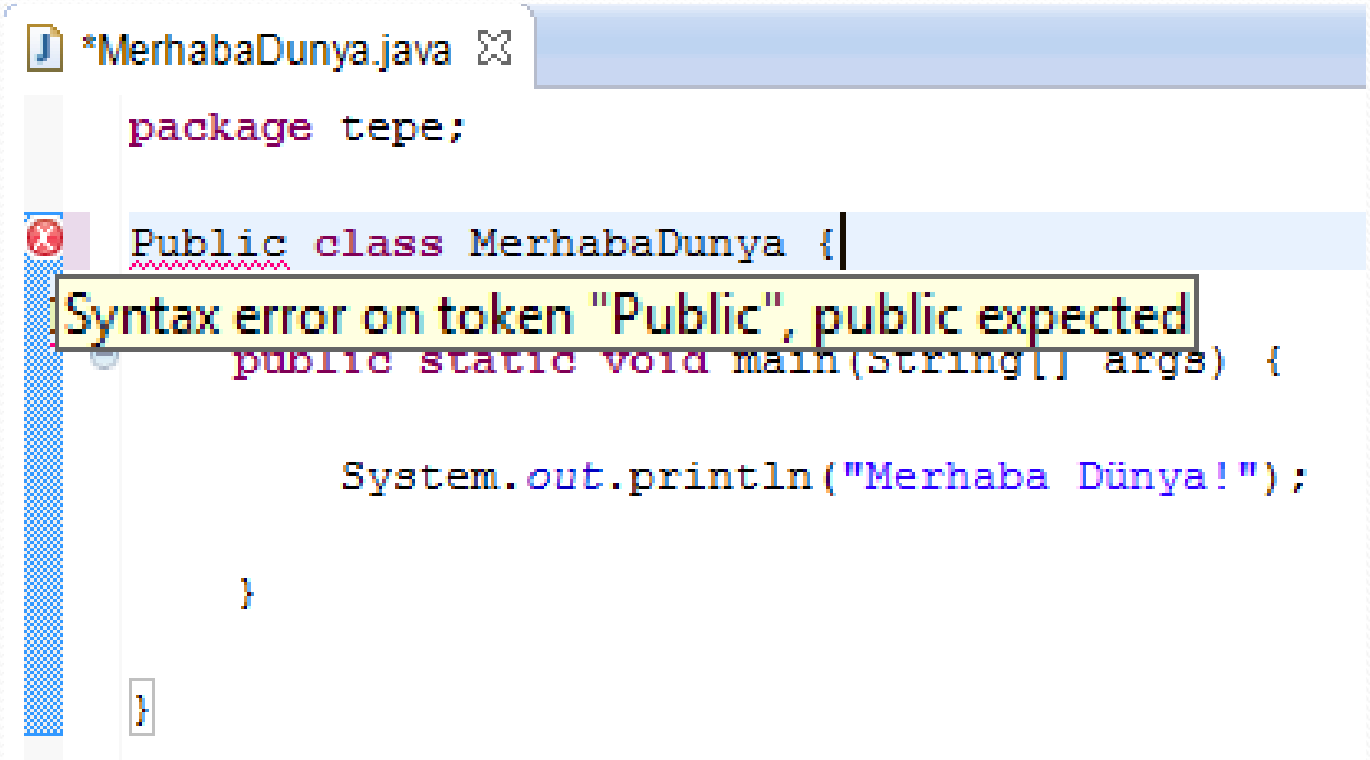
- Kod yazarken pek çok hata ile karşılaşmamız olasıdır. Bu hatalar eğer **sentaks (sözdizim) hatası** ise Eclipse bize hemen bir hata mesajı verecektir. Eğer **mantıksal bir hata** yapmış isek bunu runtime (çalıştırma) sırasında görürüz.

- **Runtime hataları** programa göre çok çeşitli olmakla birlikte **sentaks hataları** çok değişmemektedir.

- Bazı temel sentaks hatalarını gözden geçirelim.

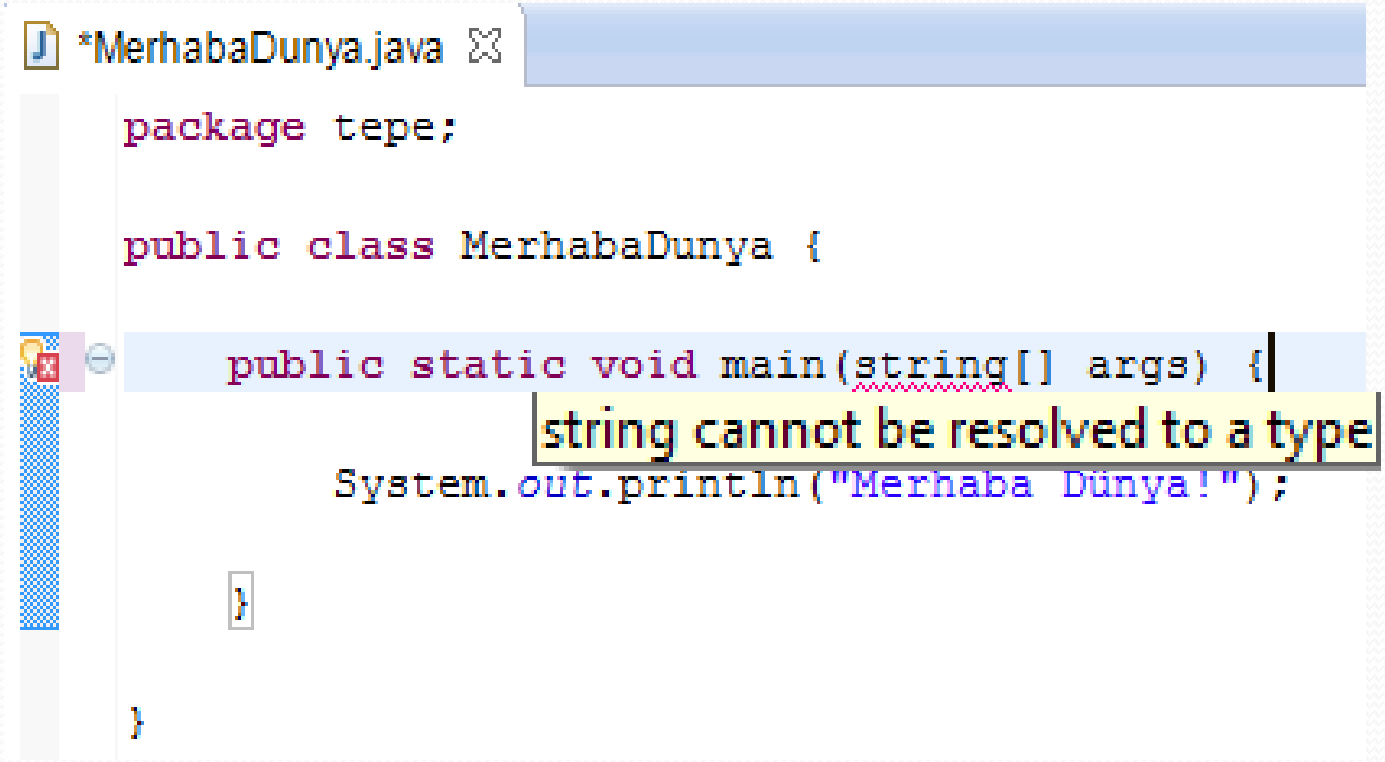
Temel Hatalar - 1

- Büyük-küçük harf hataları – 1:



Temel Hatalar - 2

- Büyük-küçük harf hataları – 2:



Temel Hatalar - 3

- Büyük-küçük harf hataları – 3:

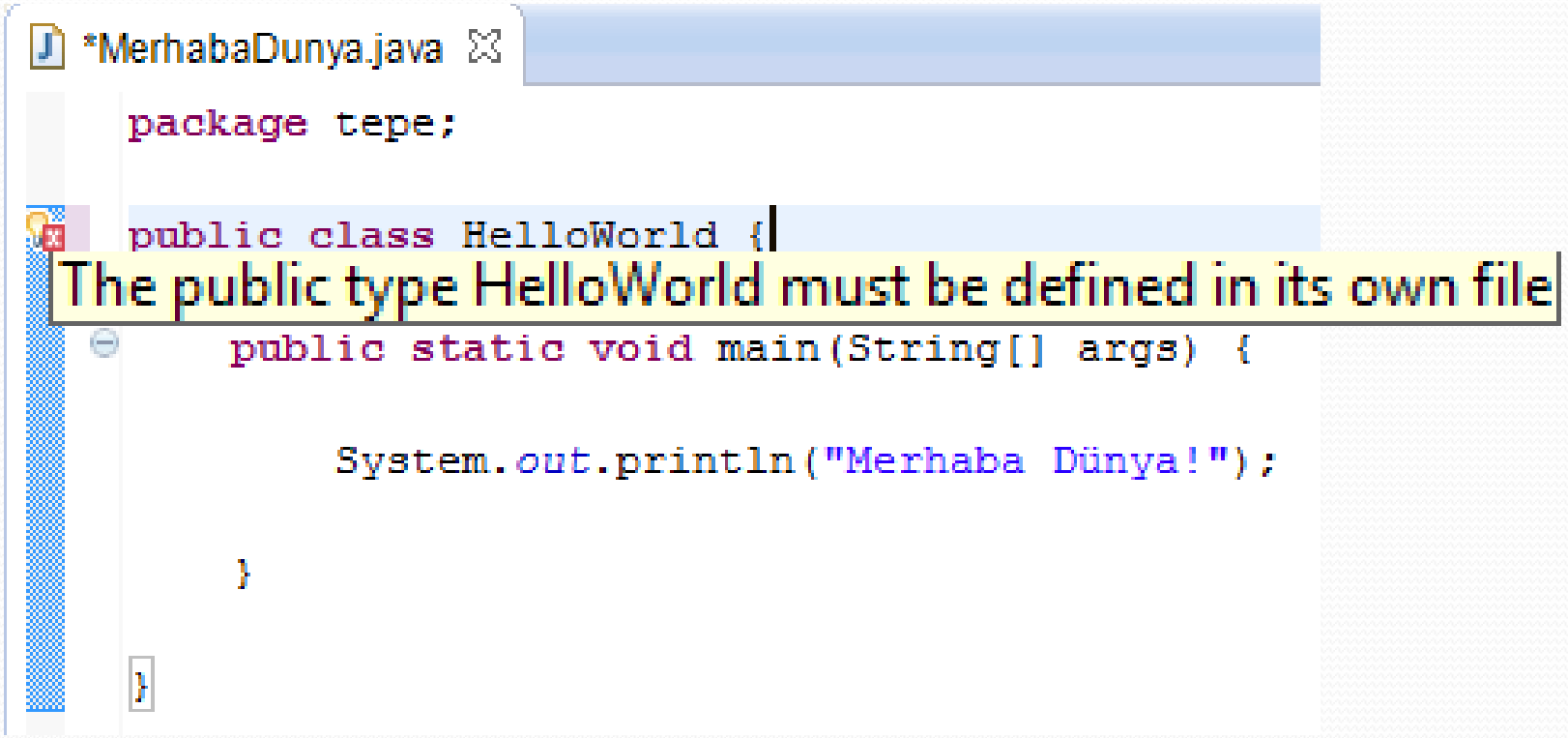
*MerhabaDunya.java

```
package tepe;

public class MerhabaDunya {

    public static void main(String[] args) {

        system.out.println("Merhaba Dünya!");
    }

}
```

system cannot be resolved

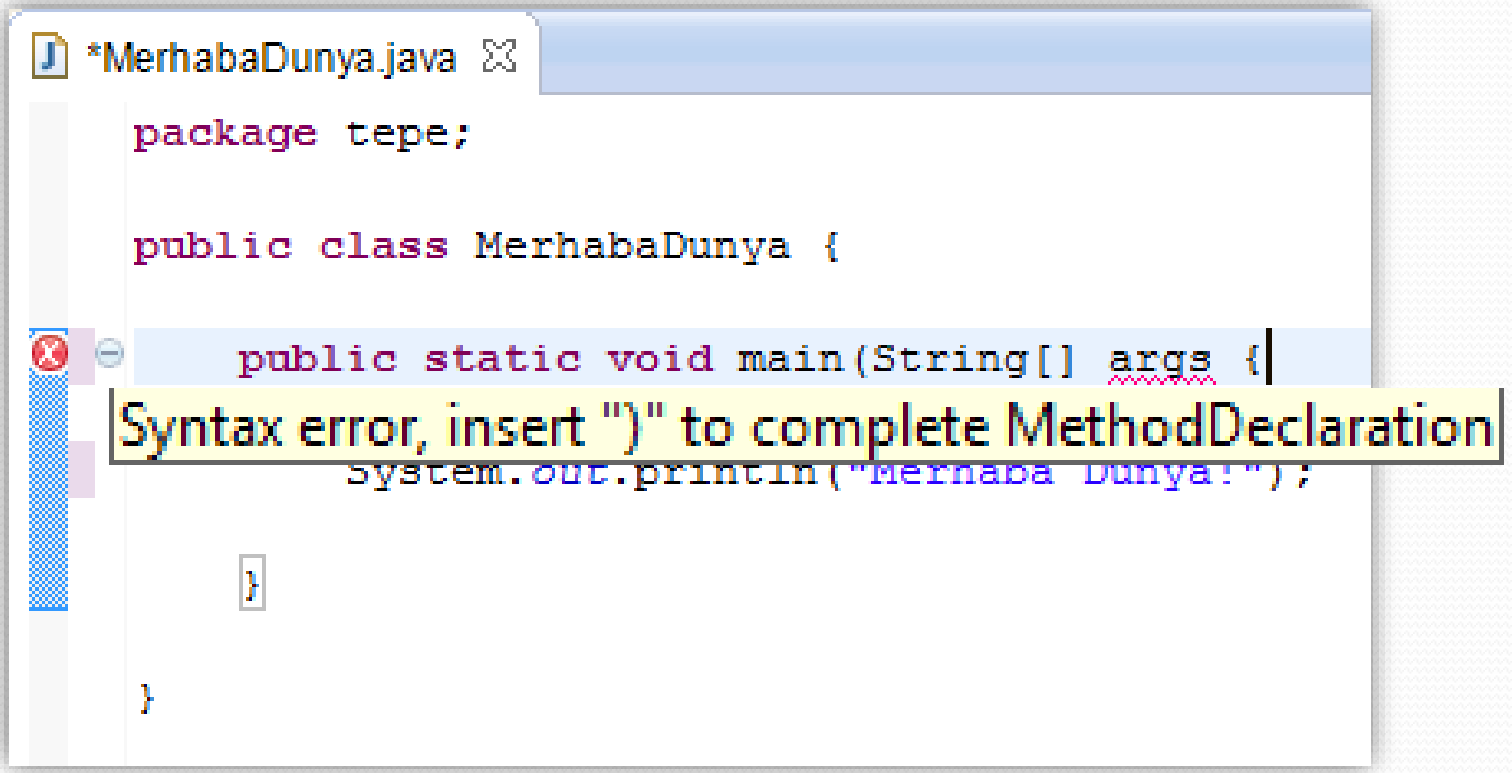
Temel Hatalar - 4

- Sınıf ismini dosya isminden farklı vermek:



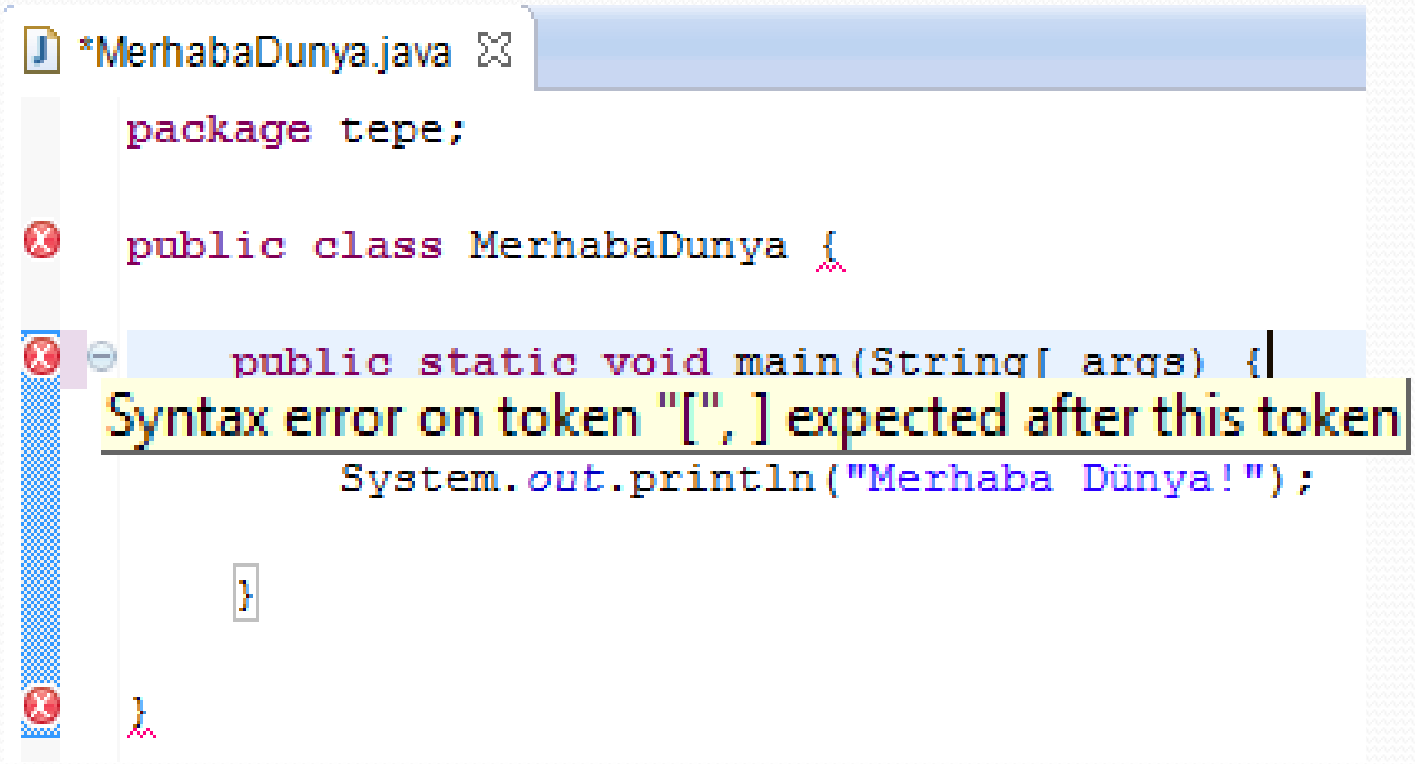
Temel Hatalar - 5

- Eksik parantez - 1:



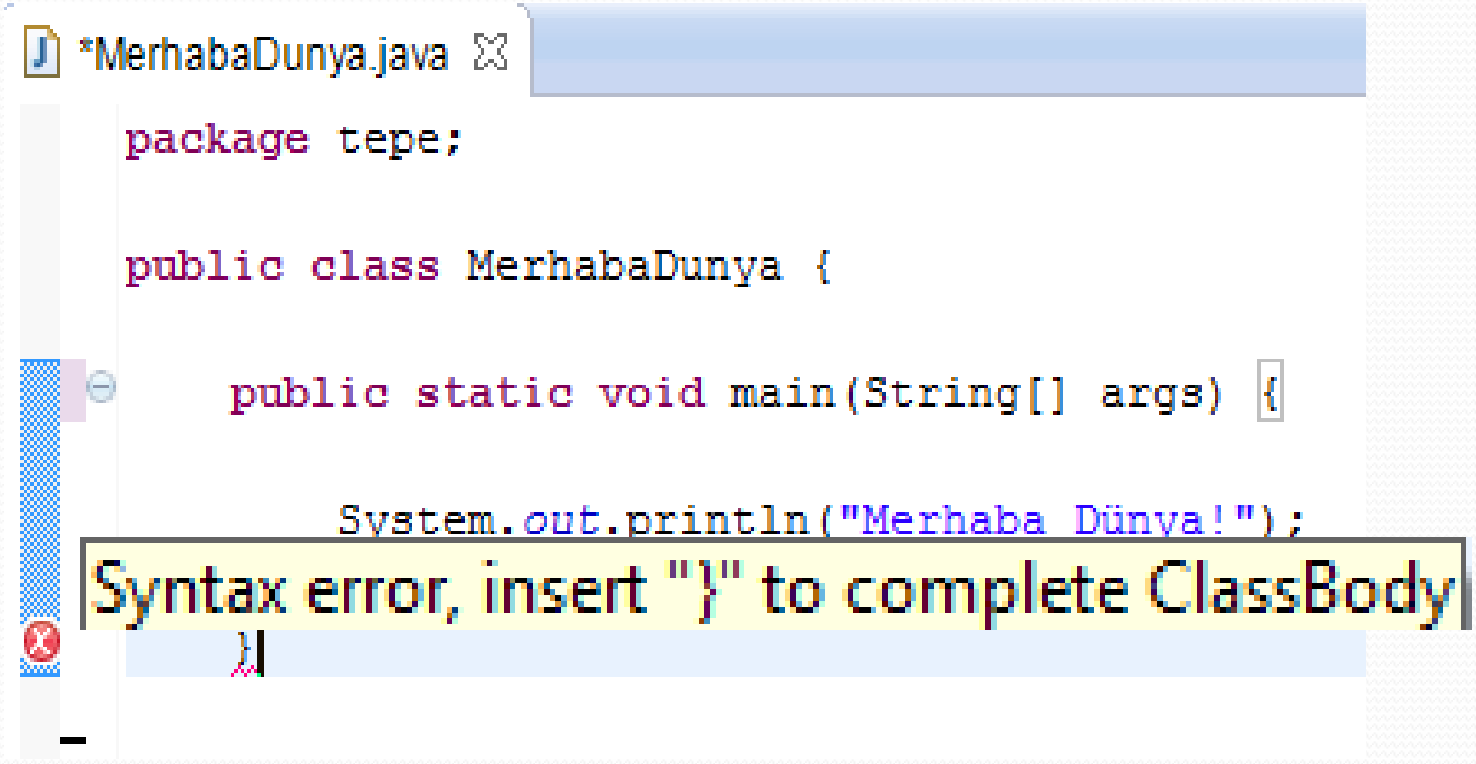
Temel Hatalar - 6

- Eksik parantez – 2:



Temel Hatalar - 7

- Eksik parantez – 3:



Temel Hatalar - 8

- Tırnak işaretlerini unutmak:

MerhabaDunya.java

```
package tepe;

public class MerhabaDunya {

    public static void main(String[] args) {

        System.out.println(Merhaba Dünya!);
```

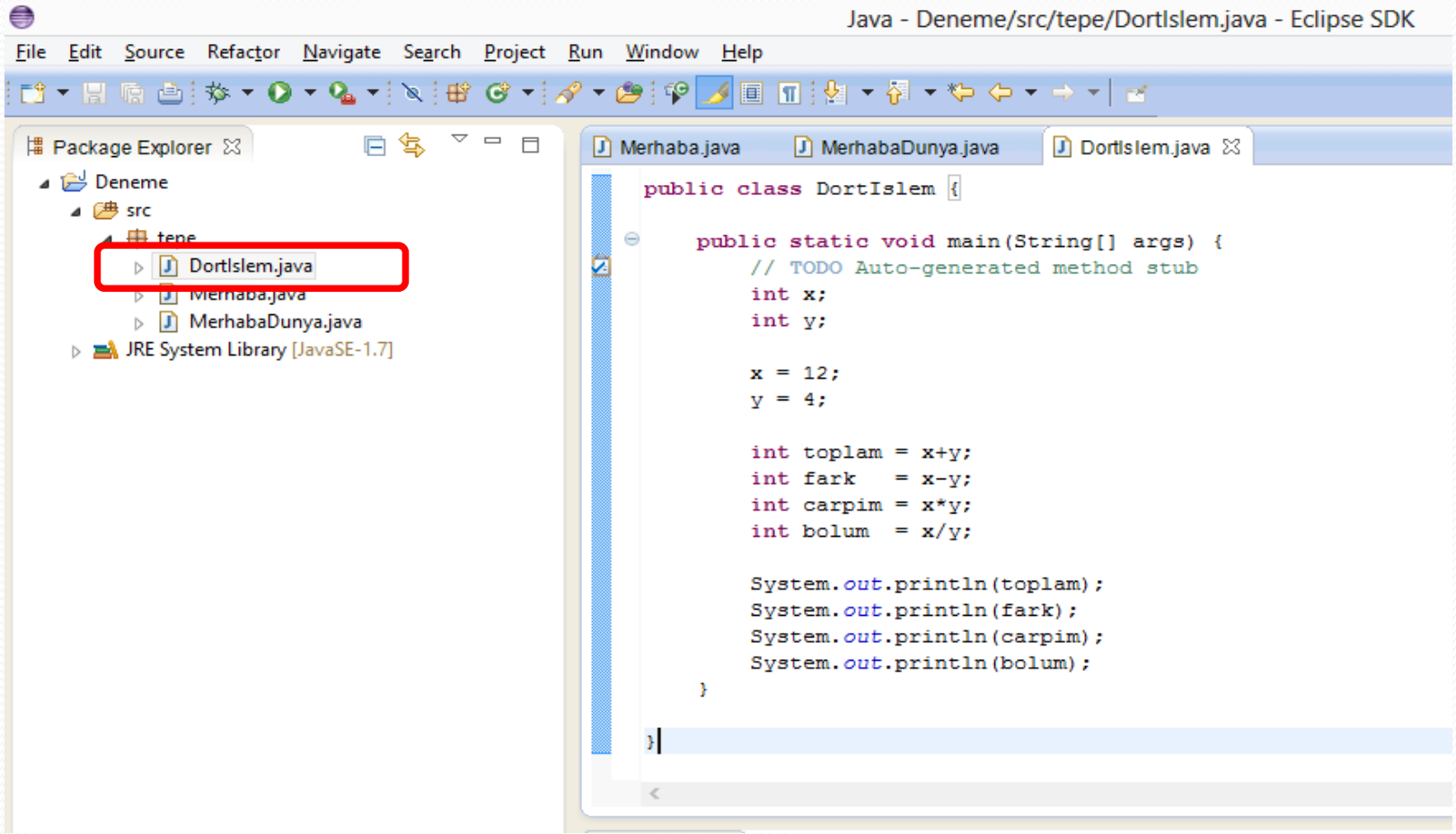
Multiple markers at this line
- Syntax error on token(s), misplaced construct(s)
- Merhaba cannot be resolved to a variable
- Syntax error, insert ";" to complete BlockStatements
- Syntax error, insert "AssignmentOperator Expression" to complete Assignment

Debug

Debug Nedir?

- **Debug** etmek, hatalardan arındırmak anlamına gelmektedir.
- Bu işlemin özelliği programın çalışması esnasında hataları görmektir.
- Debug işlemi ile programımızı adım adım çalıştırabilir, her bir adımda bellekteki değişkenleri, bunların içeriklerini vs. görebiliriz.
- Eclipse'in debug özelliğini kullanmak için basit dört işlem yapan bir program yazalım.

Debug Etmek


Java - Deneme/src/tepe/DortIslem.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Deneme
src
tepe
DortIslem.java
MerhabaDunya.java
JRE System Library [JavaSE-1.7]
Merhaba.java
MerhabaDunya.java
DortIslem.java
public class DortIslem {
public static void main(String[] args) {
// TODO Auto-generated method stub
int x;
int y;
x = 12;
y = 4;
int toplam = x+y;
int fark = x-y;
int carpim = x*y;
int bolum = x/y;
System.out.println(toplam);
System.out.println(fark);
System.out.println(carpim);
System.out.println(bolum);
}
}

Debug Etmek

Merhaba.java | MerhabaDunya.java | DortIslem.java

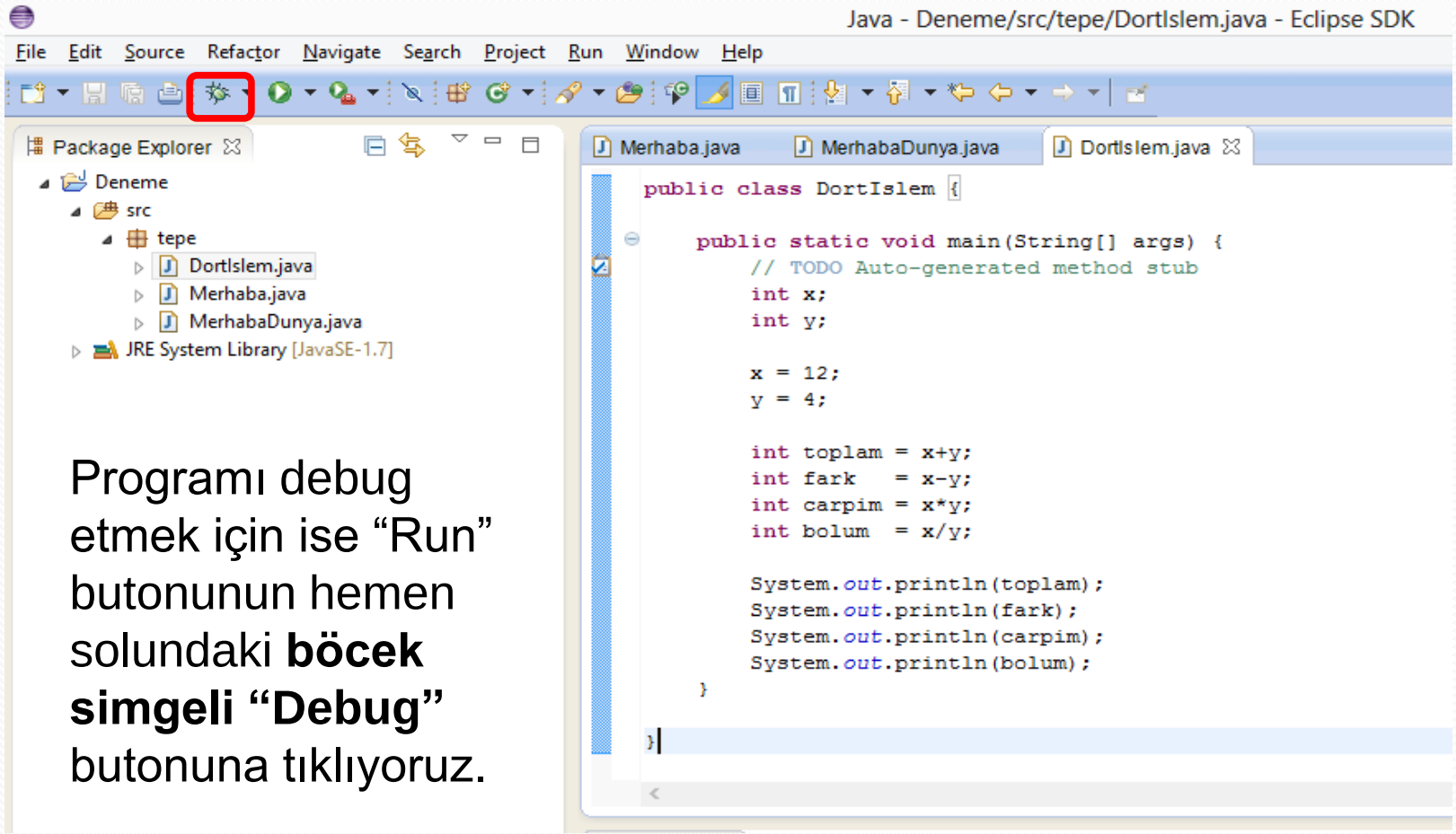
```
public class DortIslem {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int x;
        int y;

        x = 12;
        y = 4;

        int toplam = x+y;
        int fark   = x-y;
        int carpim = x*y;
        int bolum  = x/y;

        System.out.println(toplam);
        System.out.println(fark);
        System.out.println(carpim);
        System.out.println(bolum);
    }

}
```

Problems | Javadoc | Declaration | Console

```
<terminated> DortIslem [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (25 Şub 2013 19:28:17)
16
8
48
3
```

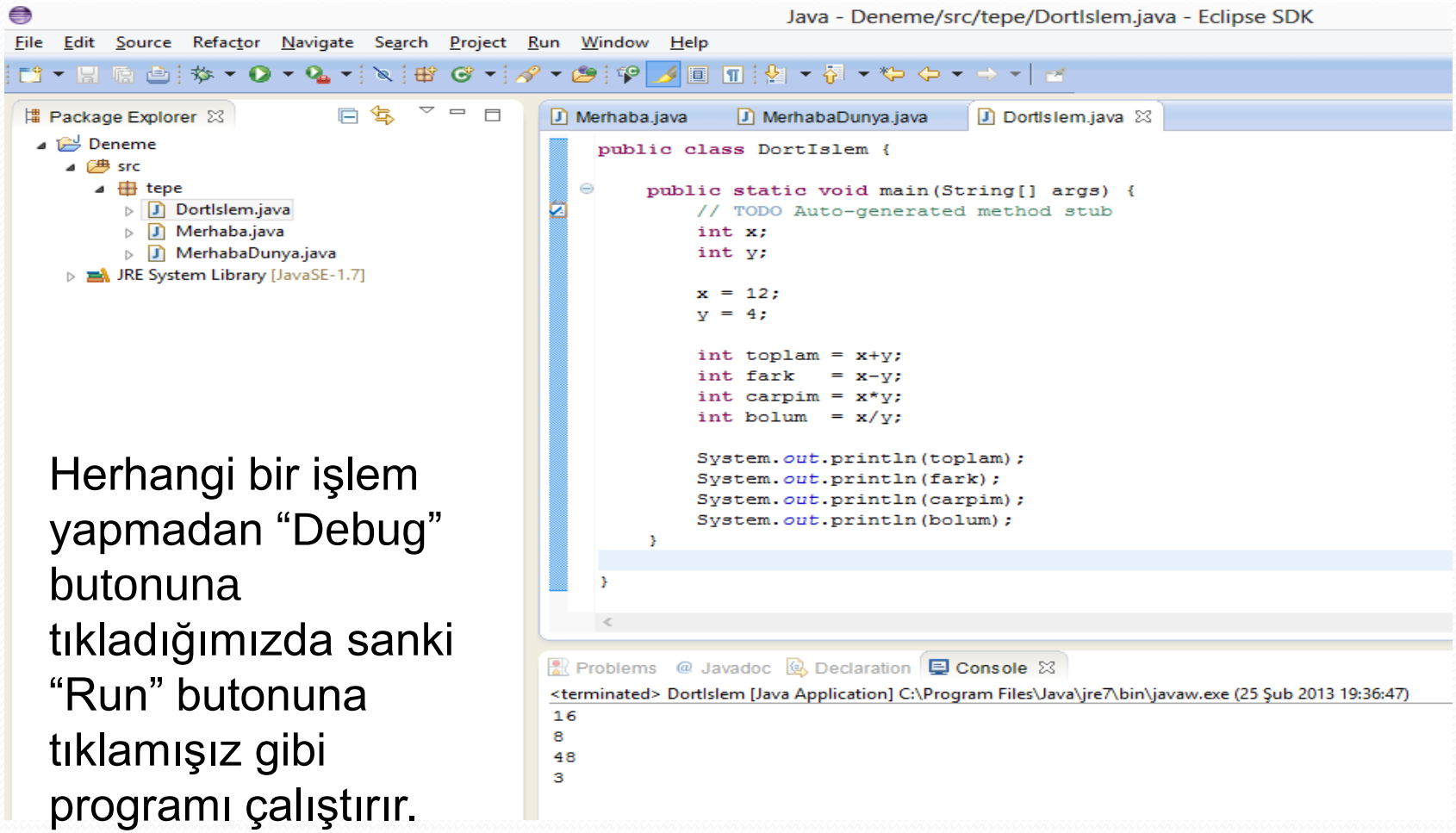
Programı normal bir şekilde "Run" ile çalıştırdığımızda yandaki gibi bir çıktı elde ederiz.

Debug Etmek



Programı debug etmek için ise “Run” butonunun hemen solundaki **böcek simgeli “Debug”** butonuna tıklıyoruz.

Debug Etmek



Herhangi bir işlem yapmadan “Debug” butonuna tıkladığımızda sanki “Run” butonuna tıklamışız gibi programı çalıştırır.

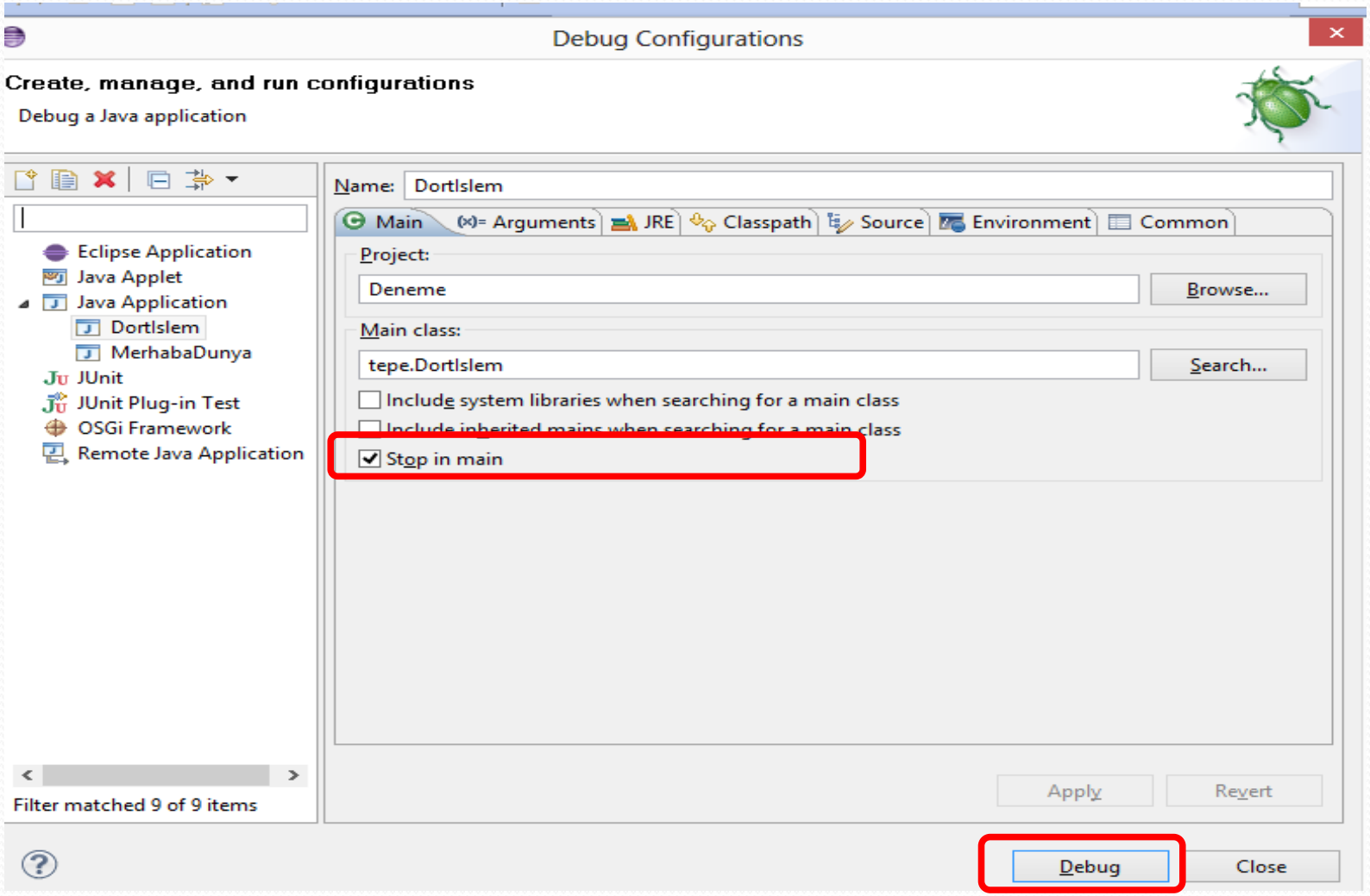
Debug Etmek

- Programı en başından itibaren debug edebilmek için küçük bir ayar yapmamız gerekiyor.

- "Debug" butonunun yanındaki oka tıklıyoruz ve **"Debug Configurations"**a giriyoruz.

- Burada **"Stop in main"** kutucuğunu işaretliyoruz. Böylece, programımız çalışmaya başladıktan sonra main() bloğuna gelince duracaktır.

Debug Etmek


Debug Configurations
Create, manage, and run configurations
Debug a Java application
Eclipse Application
Java Applet
Java Application
DortIslem
MerhabaDunya
JUnit
JUnit Plug-in Test
OSGi Framework
Remote Java Application
Filter matched 9 of 9 items
Name: DortIslem
Main
Arguments
JRE
Classpath
Source
Environment
Common
Project:
Deneme
Browse...
Main class:
tepe.DortIslem
Search...
Include system libraries when searching for a main class
Include inherited mains when searching for a main class
Stop in main
Apply
Revert
Debug
Close

Debug Etmek

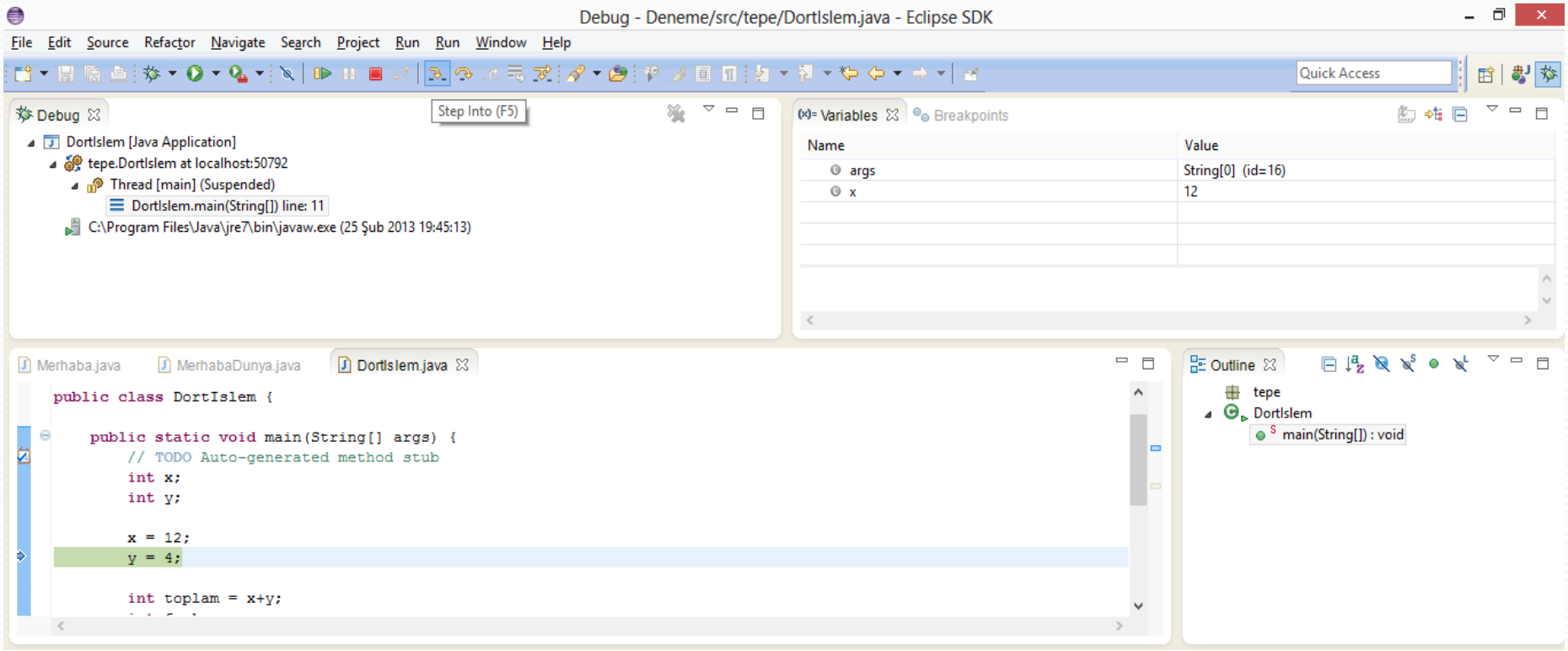
- İşlemi yaptıktan sonra "Debug" butonuna bastığımızda **perspektifimiz** değişecektir; **java perspektifinden debug perspektifine** geçeriz.

- Burada perspektif, düzen, yerleşim anlamındadır.

- Bu yeni görünümde araç çubuğunda karşımıza yeni butonlar çıkacak. Bu butonları kullanarak programımızda adım adım ilerleyebiliriz.

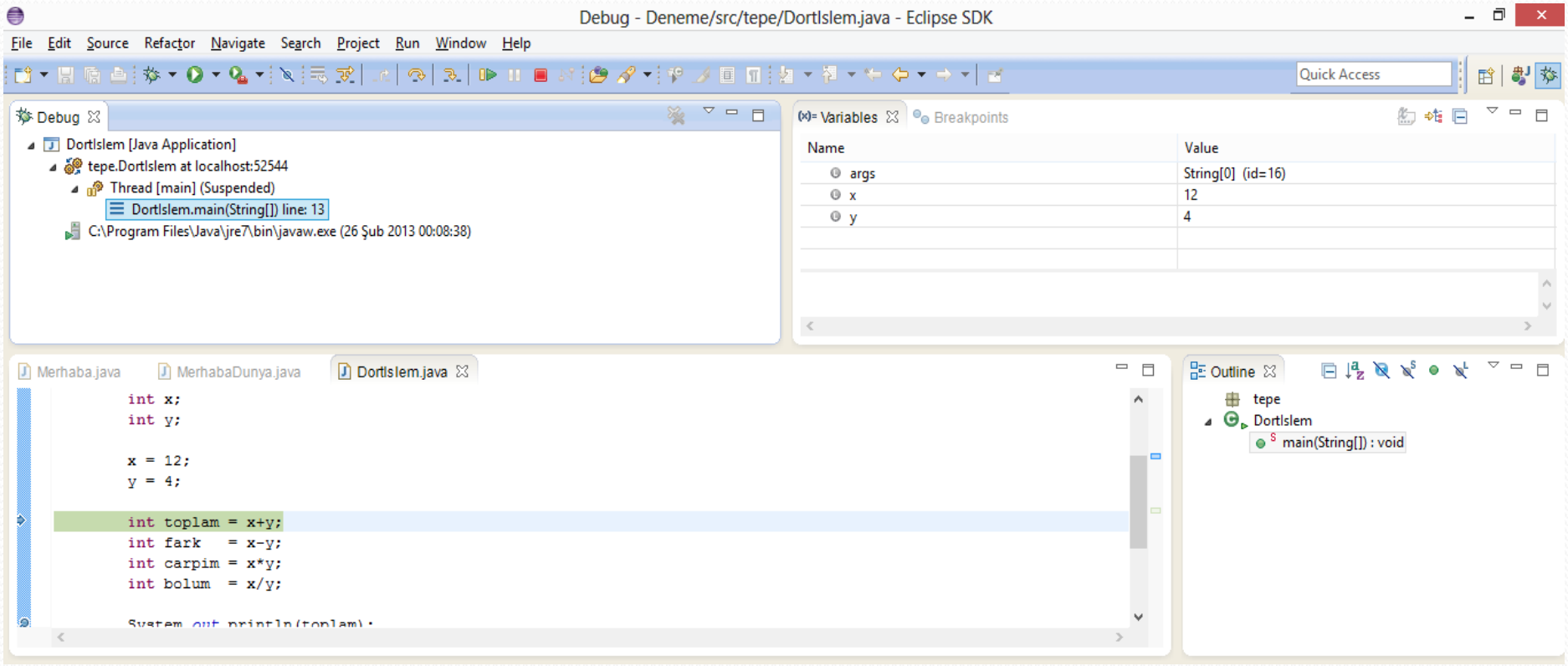
Debug Etmek

Örneğin ilk olarak F5 butonuna basarak bir satır komut çalıştıralım. Sağ üst taraftaki bölmede x değişkeninin değerinin 12 olduğunu görüyoruz.



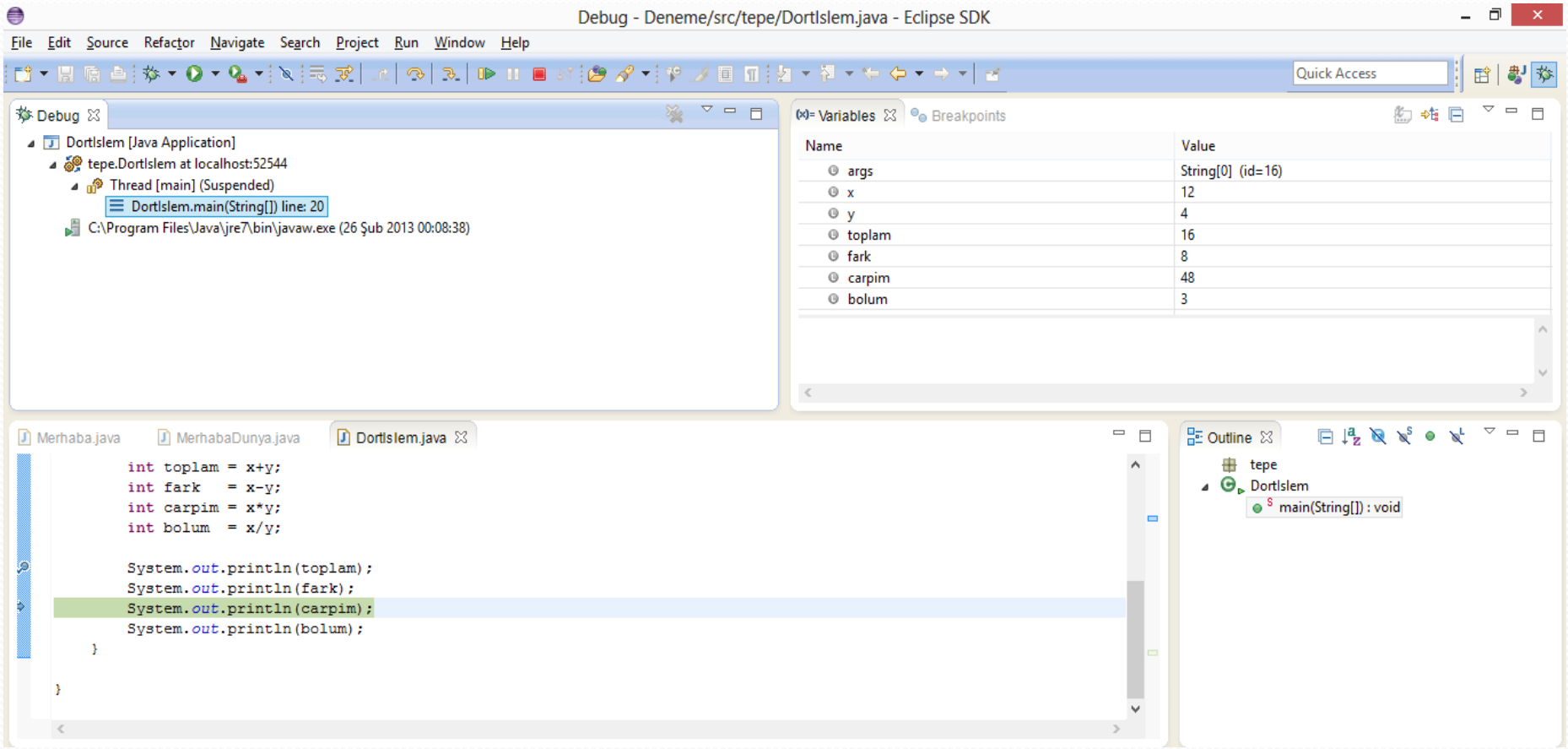
Debug Etmek

Bir sonraki adımda da y değişkeni 4 değerini alacaktır.



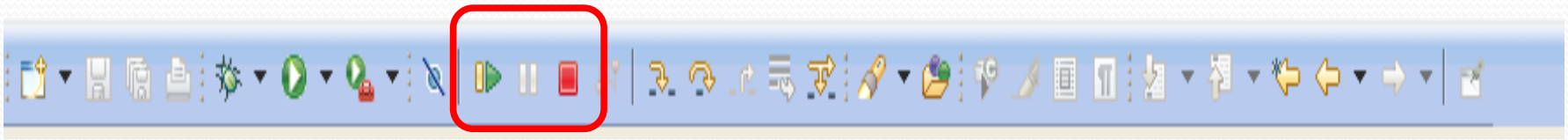
Debug Etmek

Birkaç adım sonra tüm değişkenlerin değerini görebilir hale geliyoruz.



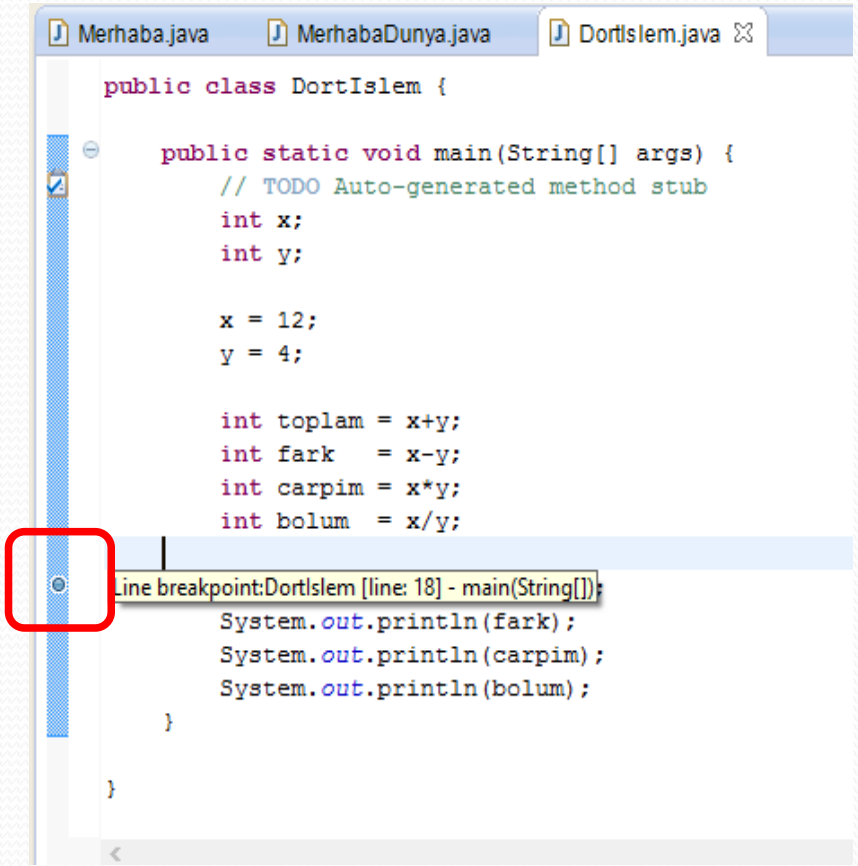
Debug Etmek

- Bu şekilde tek tek tüm değişkenlerin değerleri incelenebilir, program adım adım takip edilebilir.
- Programı çalıştırmadan sonlandırmak için **“Terminate”** butonuna tıklayabiliriz.
- Programı bir sonraki breakpoint'e kadar çalıştırmak için ise **“Resume”** butonuna tıklayabiliriz. Peki breakpoint ne demek?



Debug Etmek

- Breakpoint, programda durdurup incelemek istediğimiz yere koyduğumuz işaretlerdir diyebiliriz.
- Bir satırın solunda, pencerenin kenarına çift tıkladığımızda bu satıra bir breakpoing koymuş oluyoruz.
- Bunu yaptığımız zaman ilgili yerde mavi bir daire görürüz. Breakpoint'i kaldırmak için de aynı işlemi uyguluyoruz.



Debug Etmek

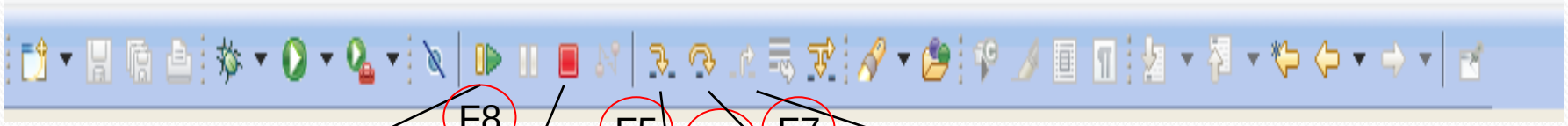
Eğer daha önce Debug işlemini main() bloğunda bekleyecek şekilde ayarlamamış olsaydık programımızı debug eder etmez yazdırma satırına gelecektik.

Ancak daha önce bu ayarı yaptık ve programı debug ettiğimizde program main() içindeki ilk çalıştırılabilir satırda bekler. “Resume” butonuna bastığımızda ise yazdırma satırına gelir ki bu noktada tüm değişkenlerin değer almış olduğunu görebiliriz.

Tekrar “Resume” butonuna bastığımızda ise programın sonuna kadar çalışıp sonlandığını görürüz; çünkü programımıza başka breakpoint koymadık.

Debug Etmek

- Debug yaparken araç çubuğunda görünen butonlar ve önemli butonların fonksiyonları aşağıdaki gibidir:



F8

Programı bir sonraki breakpoint'e kadar çalıştır.

F5

Programı çalıştırmadan sonlandır.

F6

Tek komut çalıştır. (Adım adım ilerle. Metod/döngü varsa içine gir.)

F7

Tek satır çalıştır. (Satır satır ilerle. Metod/döngü varsa içine girmeden devam et.)

Ana programa dön. (Metodun/döngünün sonucunu dön ve çık.)