

Paketi i jedinice prevodenja

Objektno-orjentisano programiranje 1



Paketi i jedinice prevođenja

- **Definicije referencijalnih tipova (klase, interfejse, enume) možemo grupisati po paketima**
 - Svaka klasa/interfejs/enum pripada nekom paketu
 - U isti paket smeštamo one tipove koji zajedno čine neku logičku celinu
 - Mehanizam za hijerarhijsko uređivanje komponenti softverskog sistema (**paket može sadržati druge pakete**)
 - Mehanizam za kontrolu konflikta imena (**različiti paketi mogu sadržati definicije tipova istog imena, u jednom paketu imena moraju biti različita**)
- **Jednica prevođenja je jedan tekstualni fajl sa ekstenzijom java koji sadrži**
 - **Deklaraciju paketa** – navodimo kom paketu jedinica prevođenja pripada
 - **Uvozne (import) deklaracije** – deklaracije kojim uvozimo imena definisana u drugim paketima
 - **Definicija tipova** – definicije klase, interfejsa i enuma u jedinici prevođenja

Paketi i jedinice prevođenja

- Ukoliko u jedinici prevođenja ne navedemo deklaraciju paketa tada sve definicije tipova iz te jedinice prevođenja pripadaju **neimenovanom (anonimnom, podrazumevanom, default) paketu**
- Neka je **A** javni tip (klasa/interfejs/enum imena **A** deklarisan sa modifikatorom pristupa **public**)
 - **A mora biti definisan u jedinici prevođenja koja se zove A.java**
 - **Posledica:** u jednoj jedinici prevođenja možemo imati najviše jednu definiciju javnog tipa
- **Direktorijum (folder) – mehanizam fajl sistema koji nam omogućava da grupišemo fajlove i druge direktorijume**
- **Struktura paketa Java softverskog sistema korespondira stukturi direktorijuma u kojoj čuvamo izvorne/prevedene fajlove**
 - **PAKET ≡ DIREKTORIJUM**

eclipse_workspace - SANUBibAgg/src/svc/sanubib/fusion/BaseMatcher.java - Eclipse IDE

File Edit Source Refactor Refactor Navigate Search Project Run Window Help

Package Explorer

- Biber19
- SANUBibAgg
 - JRE System Library [JavaSE-12]
 - src
 - svc.sanubib.fusion
 - BaseMatcher.java
 - BibLink.java
 - BibRecord.java
 - BiFuser.java
 - CorrectionLookup.java
 - Dataset.java
 - DefaultMatcher.java
 - FusedRecord.java
 - Fuser.java
 - HTMLExport.java
 - Matcher.java
 - MatchingResult.java
 - MatchType.java
 - RefPair.java
 - svc.sanubib.fusion.fusers
 - BiFusers.java
 - M10Fuser.java
 - M20Fuser.java
 - M30Fuser.java
 - M40Fuser.java
 - M50Fuser.java
 - M60Fuser.java
 - M70Fuser.java
 - M80Fuser.java
 - M90Fuser.java
 - svc.sanubib.test
 - SimpleFusionTest.java
 - svc.sanubib.xlsagg
 - CyrillicLatinConverter.java
 - XLSAgg.java
 - Referenced Libraries
 - lib

BaseMatcher.java

```
1 package svc.sanubib.fusion;
2
3 import com.aliasi.spell.JaccardDistance;
4 import com.aliasi.spell.JaroWinklerDistance;
5 import com.aliasi.tokenizer.IndoEuropeanTokenizerFactory;
6 import com.aliasi.tokenizer.NGramTokenizerFactory;
7 import com.aliasi.tokenizer.TokenizerFactory;
8
9 /**
10  * The base class for record matching
11  *
12  * @author svc
13  */
14 public abstract class BaseMatcher implements Matcher {
15     protected static double STRONG_SIM = 0.9;
16     protected static double WEAK_SIM = 0.6;
17
18     private static String errMsg1 = "";
19
20     protected String matcherName;
21
22     public BaseMatcher(String matcherName) {}
23
24     protected TokenizerFactory tf = IndoEuropeanTokenizerFactory.INSTANCE;
25     protected TokenizerFactory ngtf = new NGramTokenizerFactory(3, 4); // 3-grams & 4-grams
26     protected JaccardDistance jaccardToken = new JaccardDistance(tf);
27     protected JaccardDistance jaccardNGram = new JaccardDistance(ngtf);
28     protected JaroWinklerDistance jaroWinkler = JaroWinklerDistance.JARO_WINKLER_DISTANCE;
29
30     protected boolean exactMatch(BibRecord r1, BibRecord r2) {}
31
32     protected boolean subexactMatch(BibRecord r1, BibRecord r2) {
33         String[] r1f = r1.getFields();
34         String[] r2f = r2.getFields();
```

Problems @ Javadoc Declaration Console


No consoles to display at this time.

Writable Smart Insert 1:1


















View



 Documents



Name	Date modified	Type	Size
 fusers	9/23/2019 8:15 PM	File folder	
 BaseMatcher	11/13/2019 3:21 PM	JAVA File	4 KB
 BibLink	9/20/2019 7:45 PM	JAVA File	2 KB
 BibRecord	9/22/2019 8:30 PM	JAVA File	2 KB
 BiFuser	9/23/2019 8:40 PM	JAVA File	3 KB
 CorrectionLookup	9/22/2019 11:25 PM	JAVA File	2 KB
 Dataset	9/22/2019 6:04 PM	JAVA File	3 KB
 DefaultMatcher	9/22/2019 11:05 PM	JAVA File	1 KB
 FusedRecord	9/22/2019 8:17 PM	JAVA File	2 KB
 Fuser	9/23/2019 7:41 PM	JAVA File	10 KB
 HTMLExport	9/23/2019 7:42 PM	JAVA File	2 KB
 Matcher	9/22/2019 11:02 PM	JAVA File	1 KB
 MatchingResult	9/22/2019 11:03 PM	JAVA File	1 KB
 MatchType	9/22/2019 4:57 PM	JAVA File	1 KB
 RefPair	9/23/2019 7:13 PM	JAVA File	1 KB

fusion

File Home Share View

Search fusion

Navigation: < > << bin > svc > sanubib > fusion >

Name	Date modified	Type
fuser_out		
novi_slajdovi		
PJ_slajdovi		
OneDrive		
This PC		
3D Objects		
Desktop		
Documents		
Downloads		
Music		
Pictures		
Videos		
Local Disk (C:)		
Local Disk (E:)		
fusers	11/13/2019 10:24 PM	File folder
BaseMatcher.class	11/13/2019 10:24 PM	CLASS File
BibLink.class	11/13/2019 10:24 PM	CLASS File
BibRecord.class	11/13/2019 10:24 PM	CLASS File
BiFuser.class	11/13/2019 10:24 PM	CLASS File
CorrectionLookup.class	11/13/2019 10:24 PM	CLASS File
Dataset.class	11/13/2019 10:24 PM	CLASS File
DefaultMatcher.class	11/13/2019 10:24 PM	CLASS File
FusedRecord.class	11/13/2019 10:24 PM	CLASS File
Fuser.class	11/13/2019 10:24 PM	CLASS File
HTMLExport.class	11/13/2019 10:24 PM	CLASS File
Matcher.class	11/13/2019 10:24 PM	CLASS File
MatchingResult.class	11/13/2019 10:24 PM	CLASS File
MatchType.class	11/13/2019 10:24 PM	CLASS File
RefPair.class	11/13/2019 10:24 PM	CLASS File

15 items

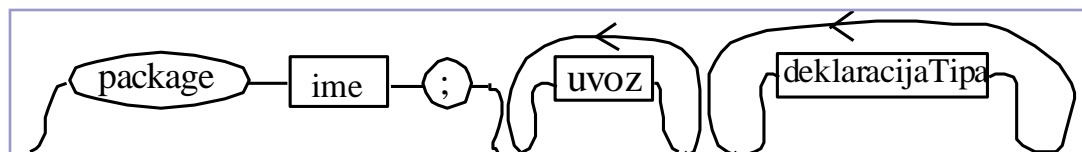
Paketi i jedinice prevođenja

- **Paketi kao mehanizam sakrivanja informacija:** tipovi (klase, interfejsi, enumi) iz paketa nisu vidljivi van paketa ukoliko nisu javni (deklarisani sa modifikatorom pristupa **public**)
- Paket može da sadrži proizvoljan broj paketa (direktorijuma) i jedinica prevođenja (fajlova)
- Paket ne mora da sadrži nijednu jedinicu prevođenja
 - Tada sadrži druge podpakete ili je prazan
- Po konvenciji imena paketa počinju malim slovima
- **Puno (kvalifikovano ime) tipa – ime koje sadrži ime tipa, ime paketa u kojem je tip definisan i imena svih nadpaketa**
 - Puno ime klase **A** koja je definisana u paketu **z**, koji se nalazi u paketu **y**, koji se nalazi u paketu **x** je **x.y.z.A**
 - Puno ime klase **B** koja je ugnježđena u klasu **A** je **x.y.z.A.B**

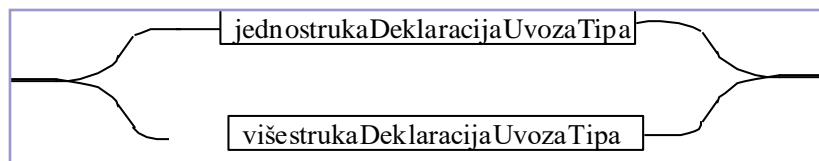
Paketi i jedinice prevođenja

- Neka je javna klasa **A** definisana u paketu **x**, i neka je klasa **B** definisana u paketu **y**
- Da bi koristili klasu **A** prilikom definisanja klase **B** tada je neophodno (**ili-ili-ili**)
 - Klasu **A** navoditi njenim punim imenom u telu klase **B**
 - U jedinici prevođenja u kojoj je definisana **B** uvesti klasu **A** (**koristimo “skraćeno” ime – samo A**)
 - U jedinici prevođenja u kojoj je definisana **B** uvesti sve klase iz paketa **x** (**takođe koristimo “skraćeno” ime**)
- **Paketi su nezavisni od podpaketa koji su sadržani u njima**
 - Klase iz podpaketa se takođe moraju uvesti ili navoditi punim imenima
 - Uvozom svih klasa iz paketa se ne uvoze klase iz podpaketa

Definicija jedinice prevođenja



Jedinica prevođenja



Uvozne deklaracije



jednostruki uvoz



višestruki uvoz

Primeri imena paketa

```
package aritmetika;  
package fizika.atomska;  
package pera.diplomski.prviDeo;
```

Primeri uvoza

```
import java.util.ArrayList;  
import java.util.*;
```

Korišćenje klase bez uvoza

```
java.util.ArrayList<Integer> v = new java.util.ArrayList<>();
```

```
// EMailList.java
```

```
package rs.uns.pmf.dmi.oop1;
```

```
import java.util.HashSet;
```

```
import java.util.Iterator;
```

```
public class EMailList {
```

```
    private HashSet<String> emails =  
        new HashSet<String>();
```

```
    public void addEmail(String email) {  
        emails.add(email);  
    }
```

```
    public void sendMessage(String msg) {  
        Iterator<String> it = emails.iterator();  
        while (it.hasNext()) {  
            String email = it.next();  
            S.o.p("Sending " + msg + " to " + email);  
        }  
    }
```

```
}
```

```
// EMailList.java
```

```
package rs.uns.pmf.dmi.oop1;
```

```
// import svih klasa iz paketa util
```

```
import java.util.*;
```

```
public class EMailList {  
    private HashSet<String> emails =  
        new HashSet<String>();  
  
    public void addEmail(String email) {  
        emails.add(email);  
    }  
  
    public void sendMessage(String msg) {  
        Iterator<String> it = emails.iterator();  
        while (it.hasNext()) {  
            String email = it.next();  
            S.o.p("Sending " + msg + " to " + email);  
        }  
    }  
}
```

```
// EMailList.java
```

```
package rs.uns.pmf.dmi.oop1;
```

```
public class EMailList {  
    private java.util.HashSet<String> emails =  
        new java.util.HashSet<String>();  
  
    public void addEmail(String email) {  
        emails.add(email);  
    }  
  
    public void sendMessage(String msg) {  
        java.util.Iterator<String> it = emails.iterator();  
        while (it.hasNext()) {  
            String email = it.next();  
            S.o.p("Sending " + msg + " to " + email);  
        }  
    }  
}
```

```
// Slajdovi.java
// u paketu lekcija5 koji je podpaket oop1

package rs.uns.pmf.dmi.oop1.lekcija5;

import rs.uns.pmf.dmi.oop1.EMailList;

public class Slajdovi {
    private EMailList list = new EMailList();

    public void dodajUcenke() {
        list.addEmail("mika@gmail.com");
        list.addEmail("zika@yahoo.com");
        list.addEmail("jova@outlook.com");
    }

    public void posaljiSlajdove() {
        list.sendMessage("Lekcija 5, slajdovi");
    }
}
```

Uvoz statičkih članova klase

```
// StaticImport.java
```

```
package rs.uns.pmf.dmi.oop1.lekcija5;
```

```
import static java.lang.System.out;
```

```
import static java.util.Arrays.sort;
```

```
public class StaticImport {  
    public static void main(String[] args) {  
        int[] arr = {7, 3, 5, 2, 1, 9, 8};  
  
        sort(arr);  
  
        for (int i = 0; i < arr.length; i++)  
            out.print(arr[i]);  
        out.println();  
    }  
}
```

Višestruki uvoz statičkih članova klase

```
// StaticImport.java
```

```
package rs.uns.pmf.dmi.oop1.lekcija5;
```

```
import static java.lang.System.*;
```

```
import static java.util.Arrays.*;
```

```
public class StaticImport {  
    public static void main(String[] args) {  
        int[] arr = {7, 3, 5, 2, 1, 9, 8};  
        sort(arr);  
  
        int pos = binarySearch(arr, 4);  
        if (pos < 0)  
            out.println("Nije u nizu");  
        else  
            out.println("Pozicija = " + pos);  
    }  
}
```

Razrešavanje kolizije imena

- Ako imamo dve klase iz različitih paketa koje se isto zovu tada
 - obe klase navodimo punim imenima
 - jednu klasu uzovimo, a jednu navodimo punim imenom

```
// VisestrukaImena.java
```

```
package rs.uns.pmf.dmi.oop1.lekcija5;
```

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

```
        com.sun.xml.bind.v2.schemagen.xmlschema.List l1;
```

```
        java.util.List<String> l2;
```

```
        java.awt.List l3;
```

```
        org.apache.xmlbeans.impl.xb.xsdschema.ListDocument.List l4;
```

```
    }
```

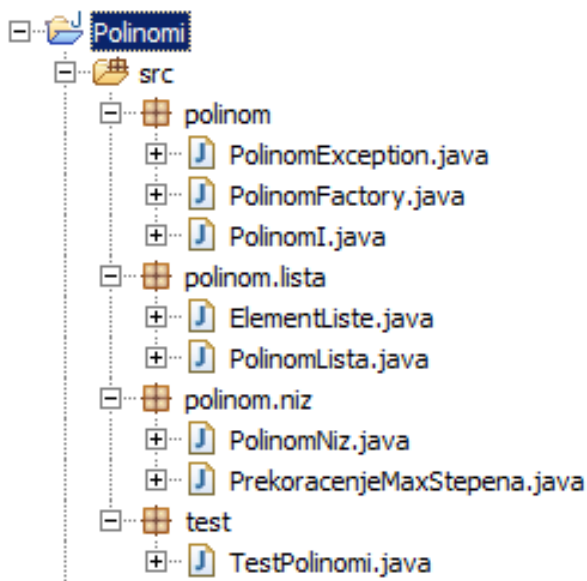
```
}
```


Imenovanje i dizajn paketa

- **Imenovanje paketa** – obrnuto ime domena firme (**rs.uns.pmf.dmi**) + ime projekta (**oop1**) + ime komponente unutar projekta (**lekcija5**)
- **Dizajn paketa prati hijerahijsku strukturu komponenti softverskog sistema**
 - Sistem se sastoji od komponenti, komponente od podkomponenti, itd.
- **“*High cohesion, low coupling*” princip**
 - **Visoka kohezivnost unutar paketa**

stepen zavisnosti klasa unutar paketa je veći nego njihov stepen zavisnosti sa klasama van paketa
 - **Slaba povezanost paketa sa drugim paketima**

lokalizovana propagacija promena (promena klase unutar paketa ne zahteva promenu klasa van paketa)
- **“*High cohesion, low coupling*” princip imamo i na nivou klasa**



```
package test;
```

```
import polinom.PolinomException;
import polinom.PolinomFactory;
import polinom.PolinomI;
```

```
public class TestPolinomi {

    public static void main(String args[])
        throws PolinomException
    {
        PolinomI polinom1 = PolinomFactory.redak();
        polinom1.dodajMonom(2, 5.0);
        polinom1.dodajMonom(1, 4.3);

        PolinomI polinom2 = PolinomFactory.redak();
        polinom2.dodajMonom(3, 2.0);
        polinom2.saberisa(polinom1);
    }
}
```

PolinomFactory sakriva podpakete paketa polinom – implementacioni detalji nebitni za korisnika

```
package polinom;
```

```
import polinom.lista.PolinomLista;
import polinom.niz.PolinomNiz;

public class PolinomFactory {
    public static PolinomI gust(int maxStepen) {
        return new PolinomNiz(maxStepen);
    }

    public static PolinomI redak() {
        return new PolinomLista();
    }
}
```

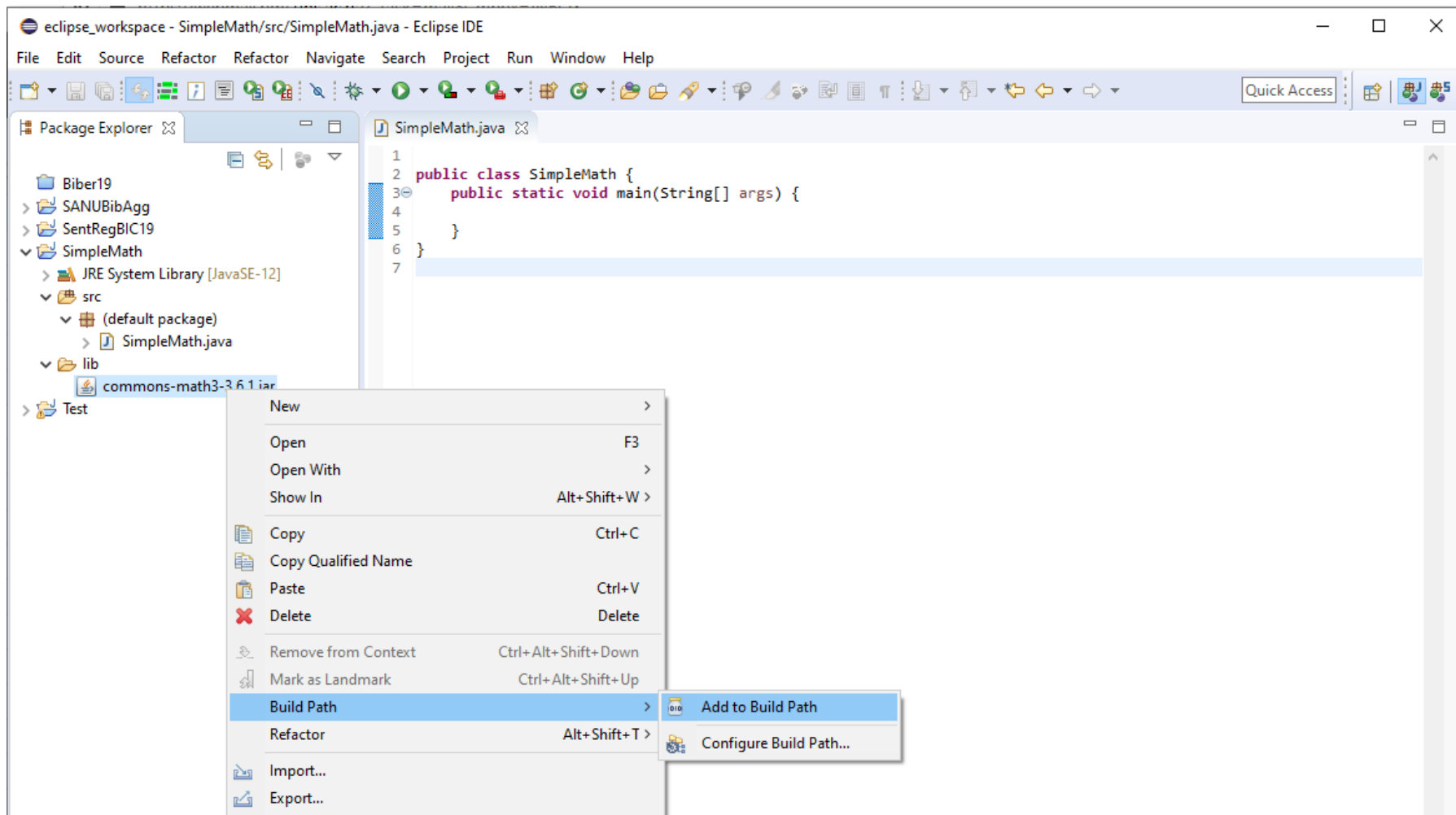
```
package polinom;
```

```
public interface PolinomI {
    void dodajMonom(int stepen, double koeficijent) throws PolinomException;
    void saberisa(PolinomI pol) throws PolinomException;
    void pomnoziSa(PolinomI pol) throws PolinomException;
    void pomnoziSa(double skalar);
    void podeliSa(PolinomI pol) throws PolinomException;
    double izracunaj(double tacka);
    double koeficijent(int stepen);
    int najveciStepen();
}
```

PolinomLista implements PolinomI
PolinomNiz implements PolinomI

Jar fajlovi

- **jar fajl – arhiva (kompresovani fajl) koji sadrži class fajlove**
- Java biblioteke se distribuiraju kao jar fajlovi



eclipse_workspace - SimpleMath/src/SimpleMath.java - Eclipse IDE

File Edit Source Refactor Refactor Navigate Search Project Run Window Help

Quick Access

Package Explorer

- Biber19
- SANUBibAgg
- SentRegBIC19
- SimpleMath
 - JRE System Library [JavaSE-12]
 - src
 - (default package)
 - SimpleMath.java
 - Referenced Libraries
 - commons-math3-3.6.1.jar
 - lib
 - commons-math3-3.6.1.jar
 - Test

SimpleMath.java

```
1 import org.apache.commons.math3.random.MersenneTwister;
2
3 public class SimpleMath {
4     public static void main(String[] args) {
5         MersenneTwister mt = new MersenneTwister();
6         System.out.println(mt.nextDouble());
7     }
8 }
9
```

Problems @ Javadoc Declaration Console

<terminated> SimpleMath [Java Application] C:\Program Files\Java\jdk-12.0.2\bin\javaw.exe (Nov 21, 2019, 1:27:52 AM)
0.1361871150700944

Writable Smart Insert 9:1

Kompajliranje iz komandne linije

● Opcija **-d**

- Iza ove opcije navodimo direktorijum koji će biti korenski za pakete navedene u jedinici prevođenja
- Npr. ako je jedinica prevođenja **Razno.java** u paketu **x.y** tada se sa **javac -d C:\mojprojekat\bin Razno.java** kreira **Razno.class** u direktorijumu **C:\mojprojekat\bin\x\y**

● Opcija **-classpath**

- Iza ove opcije navodimo direktorijume u kojima se nalaze class/jar fajlovi koje koristi klasa koju kompajliramo (**ili izvršavamo u slučaju java komande**)

Primeri korišćenja classpath opcije

```
javac -classpath .;\java\mojiPaketi;..\nekiPaketi.jar -d . NovaKlasa.java
```

● **CLASSPATH** sistemska promenljiva

Primeri korišćenja CLASSPATH sistemske promenljive

```
SET CLASSPATH=.;C:\posao\java\mojiPaketi;D:\noveStvari\nekiPaketi.jar
```