

# Java tečaj

1. dio

Uvod

# Najvažnije adrese

- ◆ Java

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

- ◆ Dokumentacija API-ja:

<http://docs.oracle.com/javase>

# Najvažnije adrese

Java SE - Downloads | Oracle Technology Network | Oracle - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Java SE - Downloads | Oracle Technology N... +


www.oracle.com/technetwork/java/javase/downloads/index.html


Oracle Technology Network > Java > Java SE > Downloads

Overview Downloads Documentation Community Technologies Training

## Java SE Downloads

Next Releases (Early Access) Embedded Use Previous Releases

  
Java Platform (JDK) 7u51  
DOWNLOAD

  
JDK 7u51 & NetBeans 7.4  
DOWNLOAD

### Java Platform, Standard Edition

#### Java SE 7u51

This release includes important security fixes. Oracle strongly recommends that all Java SE 7 users upgrade to this release.  
[Learn more](#)

#### Which Java package do I need?

- JDK:** (Java Development Kit). For Java Developers. Includes a complete JRE plus tools for developing, debugging, and monitoring Java applications.
- Server JRE:** (Server Java Runtime Environment) For deploying Java applications on servers. Includes tools for JVM monitoring and tools commonly required for server applications, but does not include browser integration (the Java plug-in), auto-update, nor an installer. [Learn more](#)
- JRE:** (Java Runtime Environment). Covers most end-users needs. Contains everything required to run Java applications on your system.

JDK  
DOWNLOAD

Server JRE  
DOWNLOAD

JRE  
DOWNLOAD

JDK 7 Docs  
Installation Instructions

Server JRE 7 Docs  
Installation Instructions


JRE 7 Docs  
Installation Instructions

#### Java SDKs and Tools

- Java SE
- Java EE and Glassfish
- Java ME
- JavaFX
- Java Card
- NetBeans IDE
- Java Mission Control

#### Java Resources

- Java APIs
- Technical Articles
- Demos and Videos
- Forums
- Java Magazine
- Java.net
- Developer Training
- Tutorials
- Java.com

  
Subscribe Today

# Najvažnije adrese

- ◆ Eclipse (trenutno Kepler)  
<http://www.eclipse.org/>  
(Eclipse IDE for Java EE Developers)


# Najvažnije adrese

Eclipse - The Eclipse Foundation open source community website. - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Eclipse - The Eclipse Foundation open source... +






www.eclipse.org

 **eclipse CON 2014**

**Java 8 is coming to EclipseCon**

SAN FRANCISCO  
MARCH 17-20, 2014  
[REGISTER NOW](#)


Visit other Eclipse Sites

mp     


Home Downloads Users Members Committers Resources Projects About Us

Google™ Custom Search

**Featured Eclipse Project**


 **EGit**

EGit is an Eclipse Team provider for the Git version control system. Git is a distributed SCM, which means every developer has a full copy of all history of every revision of the code, making queries against the history very fast and versatile. [read more](#)

**Get Started now... Download Eclipse** 

[Follow @EclipseFdn](#) [Like](#) 56k

[» Plugins](#) [» Contribute](#)  
[» Documentation](#) [» Report a Bug](#)

**Announcements** 

2014/03/04 **Countdown to EclipseCon 2014**  
EclipseCon 2014 in San Francisco is less than 2 weeks away! [Register now!](#)


2014/02/20 **Eclipse Newsletter - The Internet of Things (IoT) and Eclipse**  
Read all about the Eclipse IoT Community and its 12 projects in the latest newsletter. [Read it here.](#)

2014/02/18 **EclipseCon France 2014 - Call for Papers**  
The call for papers is now open for EclipseCon France 2014, taking place June 18-19 in Toulouse. [Propose a session.](#)

2014/02/14 **Introducing the Updated Eclipse Logo**  
A new version of the Eclipse logo is now available on the [Eclipse artwork page](#) and the [Eclipse Logo and Trademark guidelines](#) have been updated.


2014/02/05 **EclipseCon 2014 - Early Registration**  
Early registration deadline for EclipseCon 2014 is today! [Register now!](#)


[More...](#)

 **JazzHub**

Use Eclipse or Orion to develop in the cloud.

[Learn more >>](#)

**Eclipse Marketplace** 

 **Entity Designer (Juno version 3.8)**  
2011.04.05

# Najvažnije adrese

Eclipse Downloads - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Eclipse Downloads

www.eclipse.org/downloads/

**eclipse CON 2014** Starts in less than 2 weeks SAN FRANCISCO MARCH 17-20, 2014 REGISTER NOW

Visit other Eclipse Sites

Home Downloads Users Members Committers Resources Projects About Us

Google Custom Search Search

## Eclipse Downloads

Packages Developer Builds

Follow @EclipseFdn Like 56k

Eclipse Kepler (4.3.2) SR2 Packages for Windows

**Eclipse Standard 4.3.2**, 200 MB  
Downloaded 347,400 Times [Other Downloads](#)  
The Eclipse Platform, and all the tools needed to develop and debug it: Java and Plug-in Development Tooling, Git and CVS...

Windows 32 Bit  
Windows 64 Bit

### Package Solutions

Filter Packages

**Eclipse IDE for Java EE Developers**, 250 MB  
Downloaded 131,418 Times  
Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn...

Windows 32 Bit  
Windows 64 Bit

**Eclipse IDE for Java Developers**, 153 MB  
Downloaded 64,263 Times  
The essential tools for any Java developer, including a Java IDE, a CVS client, Git client, XML Editor, Mylyn, Maven integration...

Windows 32 Bit  
Windows 64 Bit

**JRebel for Eclipse IDE** Promoted Download  
See Java Code Changes Instantly. Save Time. Reduce Stress. Finish Projects Faster!

Download

**Eclipse IDE for C/C++ Developers**, 143 MB  
Downloaded 58,586 Times  
An IDE for C/C++ developers with Mylyn integration.

Windows 32 Bit  
Windows 64 Bit

**JRebel + JAVA = BACON-WRAPPED FERRARI**  
FAST & DELICIOUS CODE  
FREE TRIAL

### Related Links

- Compare & Combine Packages
- Eclipse Indigo (3.7)
- Eclipse Juno (4.2)
- Install Guide
- Documentation
- Updating Eclipse
- Forums
- Older Versions

**Hint:**  
You will need a [Java runtime environment \(JRE\)](#) to use Eclipse (Java SE 6 or greater is recommended). All

# Korisna adresa

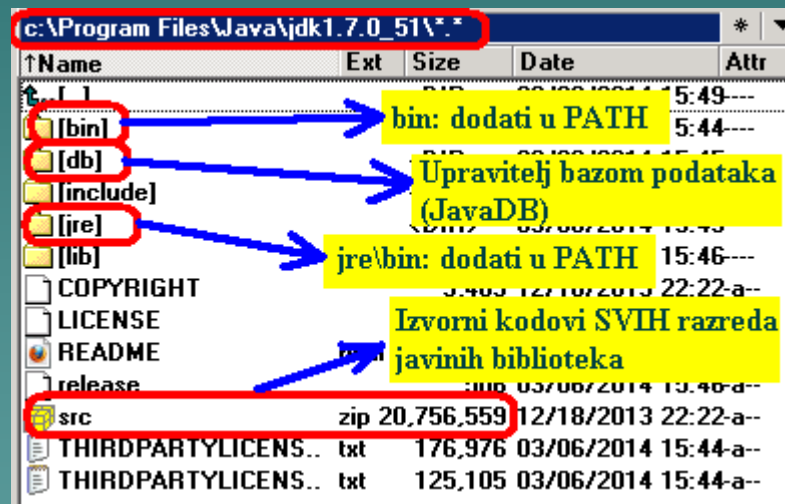
- ◆ Knjiga u nastajanju uz ovu vještinu

<http://java.zemris.fer.hr/nastava/opjj/>

- NE printati sve – poglavlja se još mijenjaju i nastajat će kako ide vještina
- U nazivu knjige gledajte najnoviji datum

# Podešavanje varijabli okruženja

- ◆ Na Windowsima JDK je tipično instaliran u Program Files\Java\Jdk...:

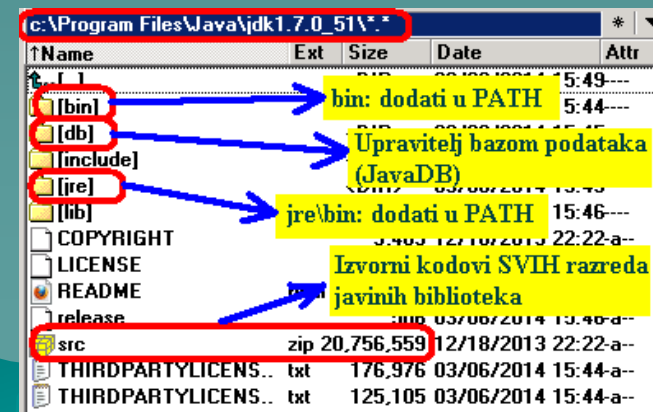


- ◆ Varijablu okruženja `JAVA_HOME` trebalo bi postaviti na taj direktorij



# Podešavanje varijabli okruženja

- ◆ Varijabla okruženja `PATH` operacijskom sustavu govori gdje da traži programe koje korisnik želi pokrenuti iz naredbenog retka
- ◆ Stoga u `PATH` treba dodati  
`%JAVA_HOME%\bin` i `%JAVA_HOME%\jre\bin`



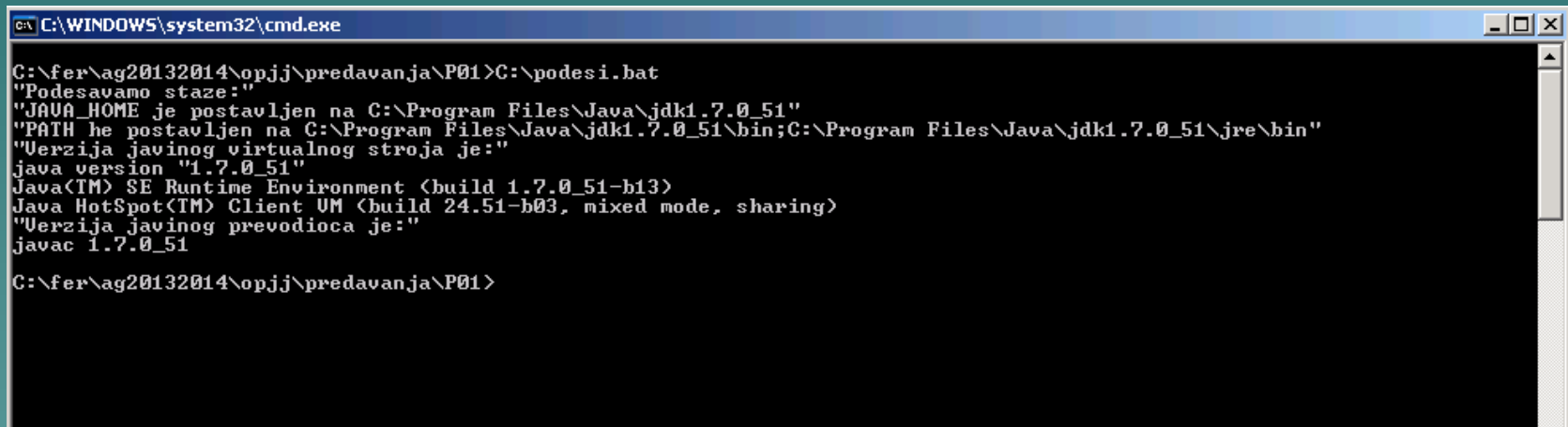
# Podešavanje varijabli okruženja

- ◆ Privremeno rješenje: napravite datoteku `podesi.bat` (u notepadu)
- ◆ Neka je npr. u `C:\podesi.bat`

```
@echo "Podesavamo staze:"  
@SET "JAVA_HOME=C:\Program Files\Java\jdk1.7.0_51"  
@SET "PATH=%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin"  
@echo "JAVA_HOME je postavljen na %JAVA_HOME%"  
@echo "PATH he postavljen na %PATH%"  
@echo "Verzija javinog virtualnog stroja je:"  
@java -version  
@echo "Verzija javinog prevodioca je:"  
@javac -version
```

# Podešavanje varijabli okruženja

- ◆ Potom otvorite konzolu (Command Prompt) i pokrenite datoteku:



```
C:\WINDOWS\system32\cmd.exe
C:\fer\ag20132014\opjj\predavanja\P01>C:\podesi.bat
"Podesavamo staze:"
"JAVA_HOME je postavljen na C:\Program Files\Java\jdk1.7.0_51"
"PATH he postavljen na C:\Program Files\Java\jdk1.7.0_51\bin;C:\Program Files\Java\jdk1.7.0_51\jre\bin"
"Verzija javinog virtualnog stroja je:"
java version "1.7.0_51"
Java(TM) SE Runtime Environment (build 1.7.0_51-b13)
Java HotSpot(TM) Client VM (build 24.51-b03, mixed mode, sharing)
"Verzija javinog prevodioca je:"
javac 1.7.0_51
C:\fer\ag20132014\opjj\predavanja\P01>
```

- ◆ Dalje nastavite raditi u toj konzoli (ako otvorite novu, ponovite proceduru i u njoj)

# Podešavanje varijabli okruženja

- ◆ Cilj je postići da kada u naredbenom retku zadate naredbe:

`java -version`

odnosno:

`javac -version`

ne dobijete poruku pogreške već  
ispisanu verziju programa

- ◆ Za detalje pogledajte u knjigu u  
dodatak.

# Podešavanje varijabli okruženja

- ◆ Ove se varijable mogu podesiti na razini operacijskog sustava (u Control Panel-u, System, Environment Variables) pa će tada vrijediti globalno
- ◆ Tada neće trebati pri svakom pokretanju konzole pokretati `podesi.bat`.

# “Hello World” program

```
package hr.fer.zemris.java.tecaj_1;
```

Naziv paketa

```
/**
```

```
 * Demonstracijski program.
```

```
 * @author Marko Cupic
```

```
 * @version 1.0
```

```
 */
```

```
public class HelloWorld {
```

Razred

```
/**
```

```
 * Metoda koja se poziva prilikom pokretanja
```

```
 * programa. Argumenti su objasnjeni u nastavku.
```

```
 * @param args Argumenti iz komandne linije.
```

```
 */
```

```
public static void main(String[] args) {  
    System.out.println("Hello World!");
```

```
}
```

```
}
```

Metoda main

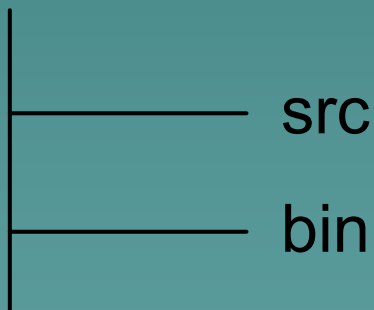
# “Hello World” program

- ◆ Otvorite *Command prompt*!
- ◆ Nađite mjesto gdje možete zapisivati datoteke (npr. C:\tecaj)
- ◆ Uđite u taj direktorij:  
C:  
cd C:\tecaj

# “Hello World” program

- ◆ Napravite dva direktorija:
  - src – tu će doći izvorni programi
  - bin – tu će doći izvršivi kod

C:\tecaj





# “Hello World” program

- ◆ Izgradite strukturu direktorija koja odgovara paketu  
hr.fer.zemris.java.tecaj\_1:

```
mkdir src\hr
```

```
mkdir src\hr\fer
```

```
mkdir src\hr\fer\zemris
```

```
mkdir src\hr\fer\zemris\java
```

```
mkdir src\hr\fer\zemris\java\tecaj_1
```

# “Hello World” program

- ◆ U *notepadu* prepisite program, i snimite ga u direktorij:  
`src\hr\fer\zemris\java\tecaj_1`  
pod nazivom:  
`HelloWorld.java`
- ◆ Pazite! Notepad zna dodati još ekstenziju `txt` pa dobijete  
`HelloWorld.java.txt` → to ne valja!

# “Hello World” program

- ◆ Uočite kako **ime datoteke mora odgovarati imenu razreda** (ono iza class), i pri tome paziti na velika i mala slova!

class HelloWorld  $\leftrightarrow$  HelloWorld.java

# “Hello World” program

## ◆ Prevođenje programa

- morate biti pozicionirani u direktoriju koji sadrži direktorije `src` i `bin`

- Zadajte:

```
javac -cp bin -sourcepath src -d bin  
src\hr\fer\zemris\java\tecaj_1\HelloWorld.java  
(sve u jednom retku!)
```

## ◆ Rezultat je `HelloWorld.class` datoteka u strukturi direktorija unutar direktorija `bin`

# “Hello World” program

- ◆ Pokretanje programa

```
java -cp bin
```

```
hr.fer.zemris.java.tecaj_1.HelloWorld
```

(sve u jednom retku!)

- ◆ Ne piše se `.class` ekstenzija

- ◆ Navodi se puno ime razreda

- ◆ “`-cp bin`” kaže gdje da traži `.class`  
(to je kao varijabla okruženja `PATH`  
za virtualni stroj koji izvodi program)

# “Hello World” program

- ◆ Pokretanje programa – opći format  
`java -cp staze puno.ime.razreda`  
`argumenti`  
(sve u jednom retku!)
- ◆ “-cp staze” može biti nepotreban,  
ovisno o postojanju varijable  
okruženja `CLASSPATH`

# “Hello World” program

- ◆ “Hello World” je primjer jednog (javnog) razreda (engl. **class**)
- ◆ Ime datoteke == ime tog razreda
- ◆ Razredi se organiziraju hijerarhijski u pakete – slično kao datoteke i kazala
  - **package** ključna riječ
- ◆ Metode koje pripadaju samom razredu: **static**

# “Hello World” program

- ◆ Pokretanje programa – **main** metoda  
public static void main(String[] args) {  
...  
}
- ◆ Argumenti iz komandne linije  
– String[] args → polje stringova



# Komentiranje koda

- ◆ Obični komentari
  - `/* komentar */` i `// komentar`
- ◆ Komentari iz kojih se generira dokumentacija (javadoc komentari)
  - `/** komentar */`
- ◆ Javadoc komentari za:
  - Same razrede
  - Pojedine metode

# Komentiranje koda

- ◆ Javadoc komentari sadrže oznake oblika @naziv vrijednost, npr.
  - @author ime\_autora, npr.  
@author Marko Cupic
  - @version verzija\_razreda, npr.  
@version 1.0
  - @param ime\_argumenta opis  
@param x broj čiji sinus treba izračunati
  - @return opis  
@return vraća sinus zadanog broja

# Komentiranje koda

```
package hr.fer.zemris.java.tecaj_1;
```

```
/**
```

```
 * @author Marko Cupic
```

```
 * @version 1.0
```

```
 */
```

```
public class HelloWorld {
```

```
    /**
```

```
     * Metoda koja se poziva prilikom pokretanja
```

```
     * programa. Argumenti su objasnjeni u nastavku.
```

```
     * @param args Argumenti iz komandne linije.
```

```
     */
```

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

```
        System.out.println("Hello World!");
```

```
    }
```

```
}
```

Ispis na ekran (.out), ili na izlaz  
za pogreške (.err)  
Postoji i printf funkcija!

# Komentiranje koda

```
/**  
 * Metoda računa y-tu potenciju od broja x.  
 * @param x argument x  
 * @param y argument y; mora biti nenegativan  
 * @return vraća iznos izraza  $x^y$   
 */  
public static double pow(int x, int y) {  
    ...  
}  
}
```

# Tipovi varijabli

Primitivni	Objektni omotači (wrappers)	Zauzeće
byte	Byte	1 oktet / ?, signed
short	Short	2 okteta / ?, signed
int	Integer	4 okteta / ?, signed
long	Long	8 okteta / ?, signed
char	Character	2 okteta / ?, UTF-16
-	String	?
boolean	Boolean	1 bit / ?
float	Float	4 okteta / ?
double	Double	8 okteta / ?

# Pravila

- ◆ boolean: true, false (različito od 0, 1)
- ◆ if(boolean\_izraz) {...}
- ◆ Pogrešno:  

```
int v = 7;  
if(v) {...}
```
- ◆ Pogrešno:  

```
int x = 7;  
while(x) { x--; }
```

# Pravila

## ◆ Nikada:

```
double x = nestoIzracunaj(...);  
if(x==0.7) {...}
```

Ne koristiti == za usporedbe decimalnih tipova!

```
if(Math.abs(x-0.7)<1E-6) {...}
```

# Primitivni tipovi ⇔ text

- ◆ Koristiti **statičke** funkcije wrappera:

```
String sBroj = "375.83";  
double dBroj =  
    Double.parseDouble(sBroj);  
String sBroj2 =  
    Double.toString(dBroj);
```

- ◆ Pogledati javadoc za dokumentaciju



# Nekoliko jednostavnih primjera

- ◆ Napisati program koji će na zaslon ispisati argumente koje dobiva prilikom pokretanja programa

```
package hr.fer.zemris.java.tecaj_1;
```

```
/**
```

```
 * @author Marko Cupic
```

```
 * @version 1.0
```

```
 */
```

```
public class IspisArgumenata {
```

```
    /**
```

```
     * Metoda koja se poziva prilikom pokretanja
```

```
     * programa. Argumenti su objasnjeni u nastavku.
```

```
     * @param args Argumenti iz komandne linije.
```

```
     */
```

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

```
        int brojArgumenata = args.length;
```

```
        for(int i = 0; i < brojArgumenata; i++) {
```

```
            System.out.println(
```

```
                "Argument " + (i+1) + ": " + args[i]
```

```
            );
```

```
        }
```

```
    }
```

```
}
```

Svako polje  
ima svojstvo  
".length"

# Nekoliko jednostavnih primjera

- ◆ Napisati program koji će prilikom pokretanja primiti jedan argument ( $x$ ), te izračunati koliko iznosi  $e^x$  razvojem u Taylorov red
- ◆ Razvoj riješiti u zasebnoj funkciji
- ◆ Program na zaslon mora ispisati rezultat

```
package hr.fer.zemris.java.tecaj_1;
```

```
/**
```

```
 * @author Marko Cupic
```

```
 * @version 1.0
```

```
 */
```

```
public class SumaReda {
```

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

```
        ...  
    }
```

```
    private static double racunajSumu(double broj) {
```

```
        ...  
    }
```

```
}
```

```
/**  
 * Metoda koja se poziva prilikom pokretanja  
 * programa. Argumenti su objasnjeni u nastavku.  
 * @param args Argumenti iz komandne linije.  
 */
```

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

```
    if(args.length != 1) {  
        System.err.println(  
            "Program mora imati jedan argument!"  
        );  
        System.exit(1);  
    }
```

```
    double broj = Double.parseDouble(args[0]);
```

```
    System.out.println("Racunam sumu...");
```

```
    double suma = racunajSumu(broj);
```

```
    System.out.println("f(" + broj + ")=" + suma + ",");
```

```
}
```

Svako polje  
ima svojstvo  
".length"

```
/**
 * Racuna e^x razvojem u Taylorov red, prema formuli:
 *  $e^x = 1 + x + (x^2/(2!)) + (x^3/(3!)) + (x^4/(4!)) + \dots$ 
 * @param broj argument funkcije e^x
 * @return iznos funkcije u tocki x=broj dobiven kao
 *         suma prvih 10 clanova Taylorovog reda.
 */
private static double racunajSumu(double broj) {
    double suma = 0.0;
    double potencija = 1.0;
    double faktorijela = 1.0;

    suma += 1.0;

    for(int i = 1; i < 10; i++) {
        potencija = potencija * broj;
        faktorijela = faktorijela * i;
        suma += potencija/faktorijela;
    }

    return suma;
}
```

# Nekoliko jednostavnih primjera

- ◆ Napisati program koji sadrži funkciju koja prima polje double-ova, koje ispisuje na zaslon po zadanom formatu
- ◆ Napisati glavni program koji će brojeve ispisati
  - Najmanje tri mjesta za cijelobrojni dio, dva mjesta za decimalni
  - Dva + dva mjesta s obaveznim ispisom predznaka

```
package hr.fer.zemris.java.tecaj_1;
```

```
import java.text.DecimalFormat;
```

```
/**
```

```
 * @author Marko Cupic
```

```
 * @version 1.0
```

```
 */
```

```
public class FormatiraniIspisDecBrojeva {
```

```
    public static void ispis(double[] polje, String format) {
```

```
        ...
```

```
    }
```

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

```
        ...
```

```
    }
```

```
}
```



```
/**
 * Metoda na standardni izlaz ispisuje polje decimalnih
 * brojeva prema zadanom formatu.
 * @param polje polje decimalnih brojeva koje treba ispisati.
 * @param format format koji govori kako polje treba ispisati.
 * @see DecimalFormat
 */
public static void ispis(double[] polje, String format) {
    DecimalFormat formatter = new DecimalFormat(
        format
    );
    for(int i = 0; i < polje.length; i++) {
        System.out.println(
            "(" + i + "): [" +
            formatter.format(polje[i]) +
            "]"
        );
    }
}
```

```
/**
```

- \* Metoda koja se poziva prilikom pokretanja
- \* programa. Argumenti su objasnjeni u nastavku.
- \* @param args Argumenti iz komandne linije.
- \*/

```
public static void main(String[] args) {  
    double[] brojevi = new double[] {  
        3.712, 55.813, 55.816, -4.18  
    };  
    ispis(brojevi, "000.00");  
    ispis(brojevi, "+00.00;-00.00");  
}
```

# Nekoliko jednostavnih primjera

- ◆ Napisati program koji će s tipkovnice čitati decimalni broj po broj i računati njihovu sumu, sve dok se upisuju nenegativni brojevi

```
package hr.fer.zemris.java.tecaj_1;
```

```
import java.io.BufferedReader;
```

```
import java.io.IOException;
```

```
import java.io.InputStreamReader;
```

```
/**
```

```
 * @author marcupic
```

```
 * @version 1.0
```

```
 */
```

```
public class CitanjeSTipkovnice {
```

```
    public static void main(String[] args) throws  
        IOException {
```

```
        ..
```

```
    }
```

```
}
```

```
public static void main(String[] args) throws IOException {
    System.out.println("Program za računanje sume pozitivnih brojeva.");
    System.out.println("Unosite brojeve, jedan po retku.");
    System.out.println(
        "Kada unesete negativan broj, ispisat ce se suma.");

    BufferedReader reader = new BufferedReader(
        new InputStreamReader(System.in)
    );

    double suma = 0.0;
    while(true) {
        String redak = reader.readLine();
        if(redak==null) break;
        double broj = Double.parseDouble(redak);
        if(broj<0) break;
        suma += broj;
    }

    System.out.print("Suma je: ");
    System.out.println(suma);

    reader.close();
}
```

## ◆ Za napredniju obradu ulaza postoji razred `java.util.Scanner`

```
Scanner sc = new Scanner(System.in);  
int i = sc.nextInt();
```

---

```
Scanner sc = new Scanner(new File("myNumbers"));  
while (sc.hasNextLong()) {  
    long aLong = sc.nextLong();  
}
```

---

```
String input = "1 fish 2 fish red fish blue fish";  
Scanner s = new  
Scanner(input).useDelimiter("\\s*fish\\s*");  
System.out.println(s.nextInt());  
System.out.println(s.nextInt());  
System.out.println(s.next());  
System.out.println(s.next());  
s.close();
```

- ◆ Za naprednije generiranje izlaza postoji podrška formatiranju

```
System.out.printf("%s\n", "bla");  
System.out.format("%s\n", "bla");
```

---

```
String novi = String.format("%4$s %3$s %2$s %1$s  
%4$s %3$s %2$s %1$s",  
                           "a", "b", "c", "d");  
→ "d c b a d c b a"
```

---

- ◆ Za detalje oko formatnog stringa pogledati dokumentaciju:

<http://docs.oracle.com/javase/7/docs/api/java/util/Formatter.html#syntax>

# Rad sa stringovima

- ◆ U Javi String nije polje znakova terminirano s `'\0'`
- ◆ Kako se točno String pohranjuje – nije nas briga
- ◆ Zahvaljujući tome, Java omogućava lakše baratanje Stringovima
- ◆ Važno: stringovi su nepromijenjivi (immutable); metode koje nešto mijenjaju vraćaju nove stringove!



```
package hr.fer.zemris.java.tecaj_1;
```

```
/**
```

```
 * @author Marko Cupic
```

```
 * @version 1.0
```

```
 */
```

```
public class RadSaStringovima {
```

```
    /**
```

```
     * Metoda koja se poziva prilikom pokretanja
```

```
     * programa. Argumenti su objasnjeni u nastavku.
```

```
     * @param args Argumenti iz komandne linije.
```

```
     */
```

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

```
        ispis1();
```

```
        ispis2();
```

```
        ispis3();
```

```
        ispis4();
```

```
    }
```

```
}
```

```
/**  
 * Demonstracija zbrajanja stringova.<br>  
 * Zbrajanje uporabom operatora + kroz vise naredbi.  
 * Vrlo neefikasno!  
 */  
private static void ispis1() {  
    String tekst = null;  
  
    tekst = "The quick " + "brown ";  
    tekst += "fox jumps over ";  
    tekst += 3;  
    tekst += " lazy dogs.";   
  
    System.out.println(tekst);  
}
```

```
/**  
 * Demonstracija zbrajanja stringova.<br>  
 * Zbrajanje operatorom + u jednoj naredbi. Efikasnije.  
 */  
private static void ispis2() {  
    String tekst = null;  
  
    int broj = 3;  
  
    tekst = "The quick brown fox jumps over " +  
           broj + " lazy dogs.";   
  
    System.out.println(tekst);  
}
```

```
/**
 * Demonstracija zbrajanja stringova.<br>
 * Zbrajanje uporabom StringBuffer objekta. Jednako efikasno
 * kao i primjer 2? Inicijalno se stvara spremnik
 * velicine 16 koji se tri puta realocira kako bi se prosirio.
 * Napomena: prije Java 5.0 koristio se StringBuffer koji je bitno
 * sporiji (ali je višedretveno siguran).
 */
private static void ispis3() {
    String tekst = null;

    StringBuilder sb = new StringBuilder();

    sb.append("The quick ").append("brown ");
    sb.append("fox jumps over ").append(3);
    sb.append(" lazy dogs.");

    tekst = sb.toString();

    System.out.println(tekst);
}
```

```
/**
 * Demonstracija zbrajanja stringova.<br>
 * Zbrajanje uporabom StringBuffer objekta. Najefikasnije
 * ako unaprijed znamo potrebnu velicinu spremnika. U primjeru
 * se alocira spremnik velicine 50 znakova.
 * Napomena: prije Java 5.0 koristio se StringBuffer koji je bitno
 * sporiji (ali je višedretveno siguran).
 */
private static void ispis4() {
    String tekst = null;
    StringBuilder sb = new StringBuilder(50);

    sb.append("The quick ").append("brown ");
    sb.append("fox jumps over ").append(3);
    sb.append(" lazy dogs.");

    tekst = sb.toString();

    System.out.println(tekst);
}
```

# Uporaba "struktura" podataka

- ◆ U ovom primjeru tretirat ćemo Javu kao C-oliki jezik; baš kao što C ima `struct` za strukture, u Javi možemo koristiti `class` za simulaciju takvog ponašanja

# Uporaba "struktura" podataka

- ◆ Tada umjesto C-ovskog alociranja:

```
struct x *var = (struct x*)malloc(  
    sizeof(struct x)  
);
```

u Javi pišemo:

```
x var = new x();
```

# Uporaba "struktura" podataka

- ◆ Razmotrit ćemo jednostavan primjer izgradnje jednostavnog stoga
- ◆ Treba podržati mogućnost dodavanja na stog, ispitivanja je li stog prazan te mogućnost skidanja elementa sa stoga
- ◆ Prikazano rješenje nije u duhu OOP-a; Javu ovdje koristimo kao da je C



```
package hr.fer.zemris.java.tecaj_1;
public class DemoStoga {

    static class Zapis {
        Zapis stari;
        String vrijednost;
    }

    public static void main(String[] args) {
        Zapis stog = null;

        stog = dodaj(stog, "Ana");
        stog = dodaj(stog, "Ivana");
        stog = dodaj(stog, "Jasna");

        while(nijePrazan(stog)) {
            Zapis vrh = stog;
            stog = ukloni(stog);
            System.out.println("Uklonio sam ime: "+vrh.vrijednost);
        }
    }

    // još implementacije metoda: sljedeći slide...
}
```

```
public class DemoStoga {  
  
    // nastavak s prethodnog slidea:  
  
    static Zapis dodaj(Zapis stog, String ime) {  
        Zapis glava = new Zapis();  
        glava.vrijednost = ime;  
        glava.stari = stog;  
        return glava;  
    }  
  
    static boolean nijePrazan(Zapis stog) {  
        return stog != null;  
    }  
  
    static Zapis ukloni(Zapis stog) {  
        return stog.stari;  
    }  
}
```