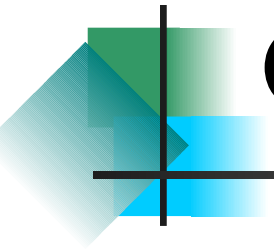
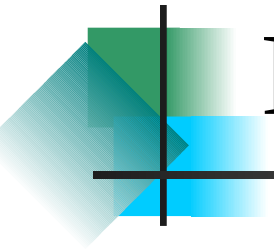


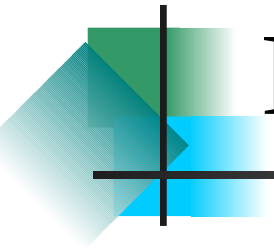
# Overview

- An overview of the course
  - What was the goal of the course?
- Where are we?
  - What did we achieve?
- Future?
  - What can we do now ...



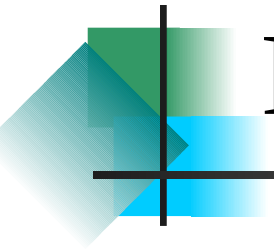
# Repetition

- AWT (Abstract Window Toolkit) or SWING
  - Allows you to create window programs
  - is the standard call set (API) for creating a graphical user interface (GUI)



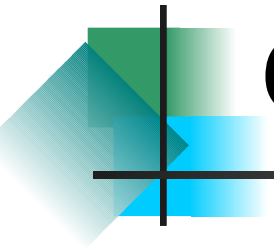
# Repetition

- Event driven programming
  - When the user interacts with the interface, the components change internal state
  - Add a handler to the components
    - An event is triggered, the handler intercepts and responds accordingly



# Repetition

- LayoutManager (ravnatelj izgleda)
  - Assists in placing components on (vsebovalnik) container
  - Examples:
    - BorderLayout
    - GridLayout
    - FlowLayout
    - GridbagLayout

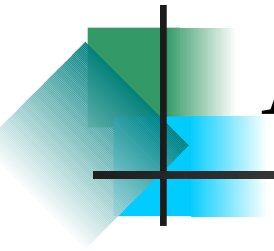


# Component: TextField / JTextField

- TextField is a single-line input field
- Constructors:
  - `TextField()`
  - `TextField(int nr_chars)`
  - `TextField(String text)`
  - `TextField(String text, int nr_chars)`
- Methods:
  - `String getText()`
  - `Void setText(String text)`
  - `...`

# Component: Menu

- Menu toolbar, menus
  - MenuBar, Menu, MenuItem
- Constructors:
  - `MenuBar()`
  - `Menu(), Menu(String text)`
  - `MenuItem(), MenuItem(String name)`
- Using the method: `add()`



# Array

- Array
  - repetition
- Multidimensional array
  - basics
  - matrices
  - usage
- Example

# Array – repetition

- Array is a string of variables of the same type
- Each element can be accesses using index

- Declaration (2 steps):
  - *type variablename*[];
  - *variablename* = new *type*[*size*];

  Or shortere (1 step):
  - *type variablename* = new *type*[*size*];

# Array - example

```
public static void main(String args[]) {
    String dan_v_tednu[] = new String[7];
    dan_v_tednu[0] = „Ponedeljek“;
    dan_v_tednu[1] = „Torek“;
    dan_v_tednu[2] = „Sreda“;
    dan_v_tednu[3] = „Cetrtek“;
    dan_v_tednu[4] = „Petek“;
    dan_v_tednu[5] = „Sobota“;
    dan_v_tednu[6] = „Nedelja“;
    System.out.println(„Drugi dan tedna: “ +
       dan_v_tednu[1]);
}
```

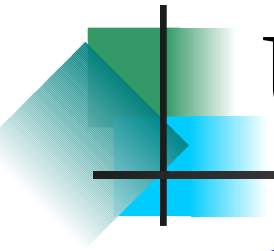
# Matrices

- Example:
  - int matrika[][] = new int[4][4];
    0 0 0 0
    0 0 0 0
    0 0 0 0
    0 0 0 0
  - matrika[1][2] = 1;
    0 0 0 0
    0 0 1 0
    0 0 0 0
    0 0 0 0

# Matrices

- We usually traverse through matrices with double *for* loop
- The first for goes by lines, the second by elements in each line.

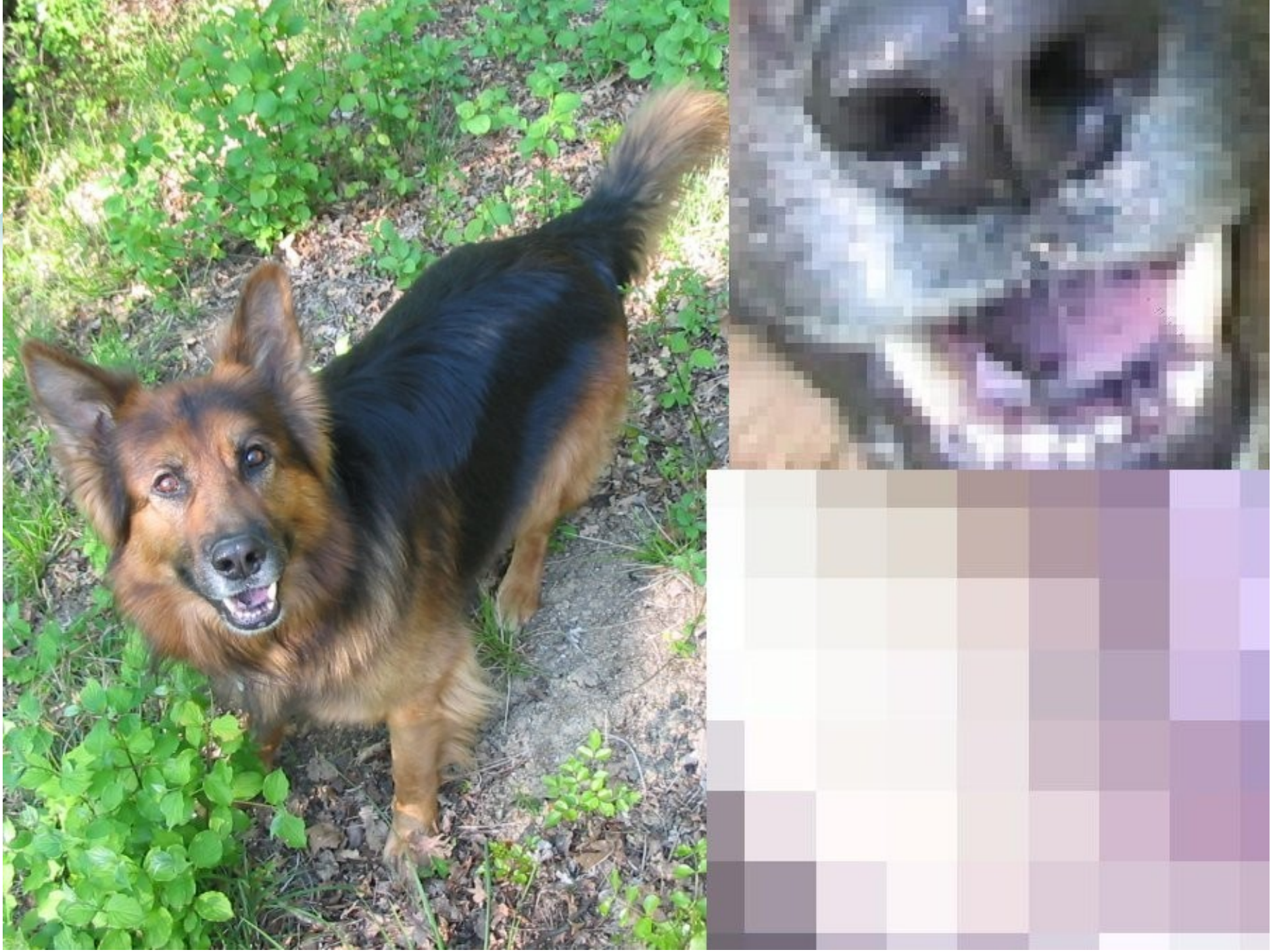
```
for(int i=0; i<4; i++){
        for(int j=0; j<4; j++){
                System.out.print(matrika[i][j] + „ “);
        }
        System.out.println();
}
```

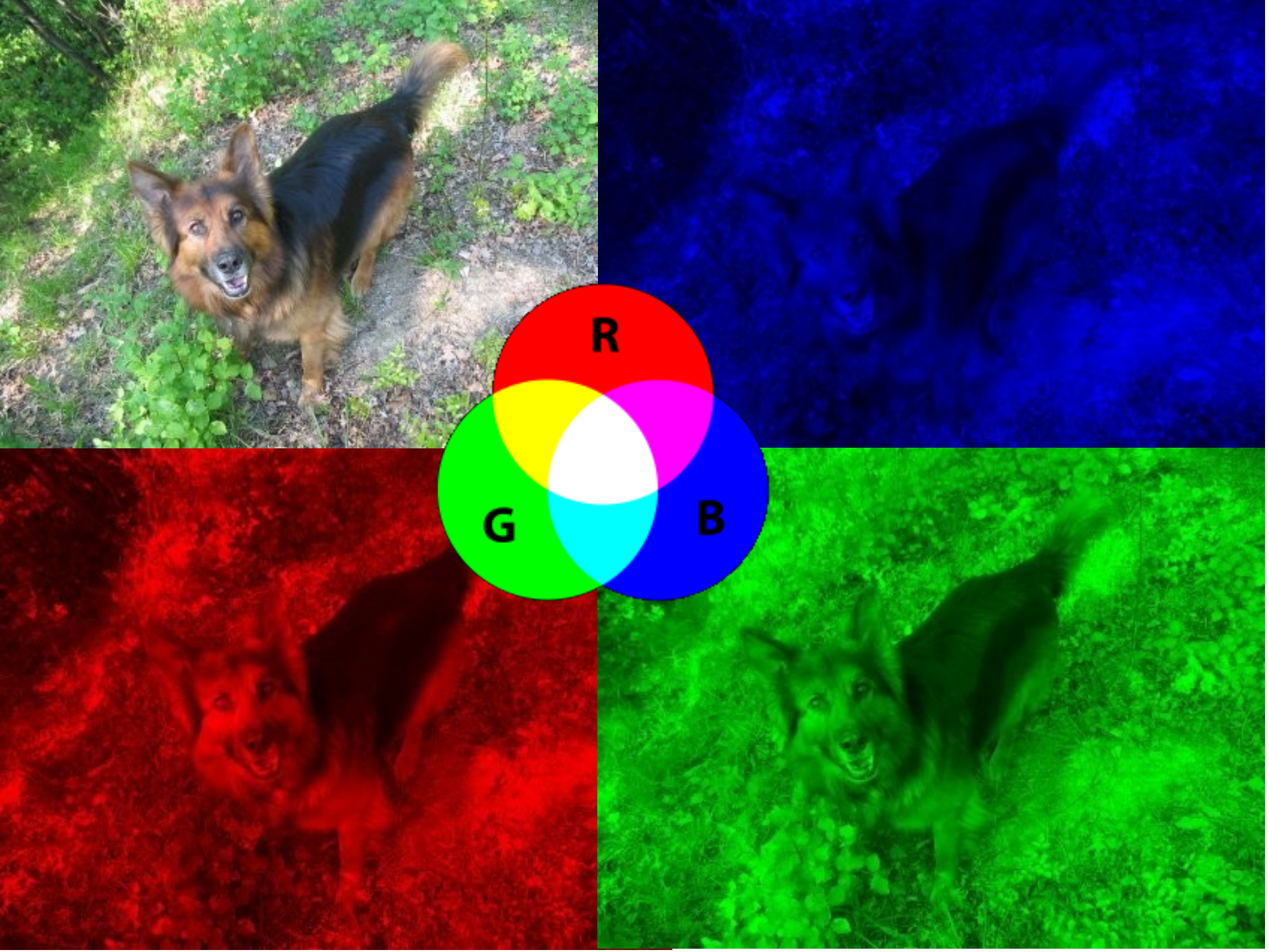


# Usage

- 2D arrays can be used to represent images
  - the value of each point in the figure is representedby a location in matrix

- An image in RGB is represented by 3 2D arrays



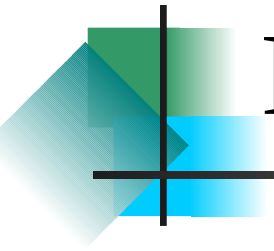

R
G
B

# Example

- B/W images
  - B/W images can be represented by a 2D array: Integer values from 0 to 255
    - 0 black, 255 white
- Example Slika2D.java

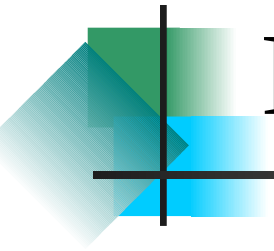
# Multidimensinal array

- An array of arrays
  - 2D – also matrix
- Declaration:
  - *type variable_name*[][] = new *type*[*size1*][size2];
- Primer:
  - int matrika[][] = new int[4][4];
- More than 2 dimensions:
  - int cube[][][] = new int[5][5][5];



# Repetition

- Class:
  - Is a „template“ for objects
  - It comprises:
    - properties/variables
    - methods/functions



# Repetition

- Example class
  - Seat:
    - properties: nr. legs, material, backrest, ...
    - methods: tvoritelj (constructor), returnNrLegs, material, ...

# Abstractions

- Example class

```
class Seat {
        int nrLegs;
        String material;
        boolean backrest;
        ...

        returnNrLegs(){
                return nrLegs;
        }
        ...
}
```

# Abstractions

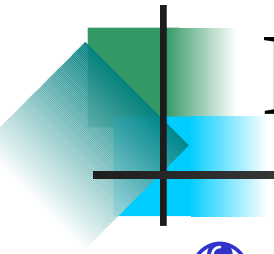
- Primer razreda

```
class Stol {
        int st_nog;
        String material;
        boolean naslonjalo;
        ...

        getSt_nog(){
                return st_nog;
        }
        ...
}
```

# Repetition

- (Predmet) Object is an implementation of a class
  - A new object is created with keyword new

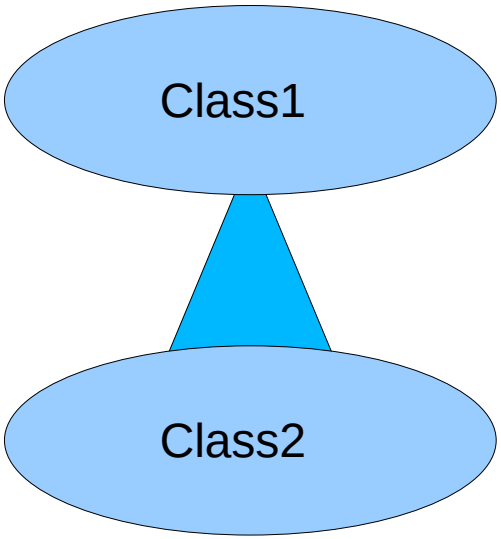


# Repetition

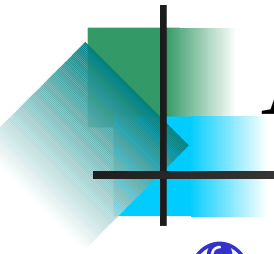
- Access to classes and methods/propertise:
  - public
  - private
  - protected

# Repetition

- Class inheritance
  - A class has subclasses and super-classes




  - A subclass inherits methods and properties from super-class

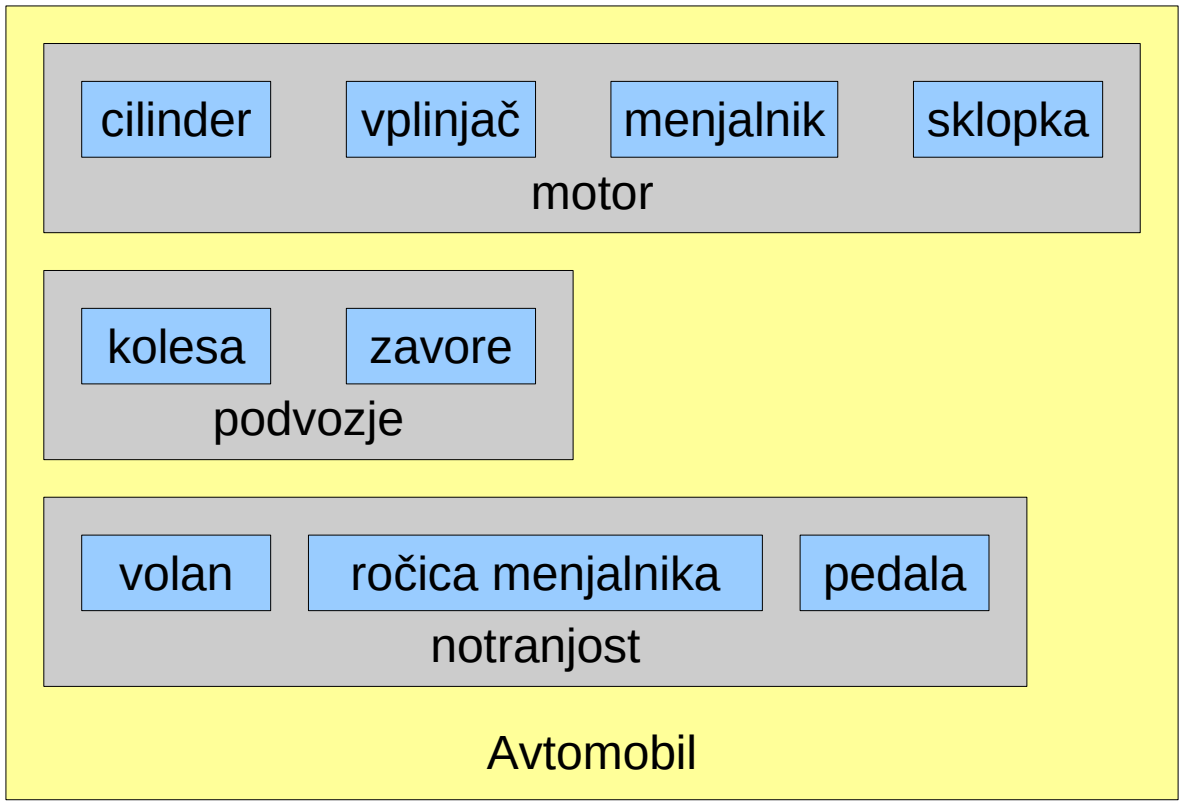


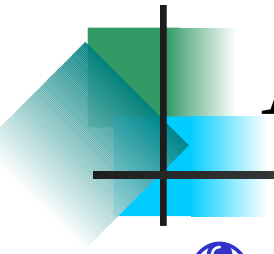
# Abstractions

- Abstraction hides complexity

# Abstractions

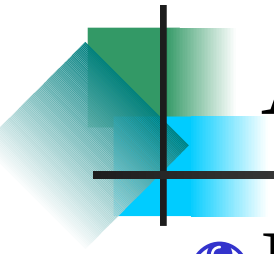
- Z abstrakcijo skrijemo kompleksnost
  - primer avtomobila:





# Abstractions

- Abstract methods and classes
  - An abstract method can be defined in Java:
    - prepare only method signature
    - Method must be implemented by a new class
  - If a class has an abstract method, then it is an abstract class



# Abstractions - image

- Example Slika2D.java