יחידה 4

מחלקות ושיטות

שלב 1- מחלקות שכל תכונותיהן הן טיפוסים פשוטים

Judy Isaacs

נתון הממשק הבא למחלקה Circle

Circle

Class Edit Tools Options

Compile Undo Cut Copy Paste Find... Find Next Close Interface

Class Circle

```
java.lang.Object
 └ Circle
```

```
public class Circle extends java.lang.Object
```

A circle that can be manipulated and drawn or erased . adapted from BlueJ example by Michael Kolling and David J. Barnes

Constructor Summary

`Circle(int diameter, int xPosition, int yPosition, java.lang.String color)` Create a new circle with x position, y position, diameter and color.

Loading class interface... Done. saved

Class Edit Tools Options

Compile | Undo | Cut | Copy | Paste | Find... | Find Next | Close

Interface

Method Summary

void	**changeColor**(java.lang.String newColor) Change the color.
void	**changeSize**(int newDiameter) Change the size to the new size (in pixels).
void	**draw**() Draw the circle with current specifications on screen.
void	**erase**() Erase the circle on screen.
void	**moveHorizontal**(int distance) Move the circle horizontally by 'distance' pixels.
void	**moveVertical**(int distance) Move the circle vertically by 'distance' pixels.

Loading class interface... Done.

saved

Circle

Class Edit Tools Options

Compile Undo Cut Copy Paste Find... Find Next Close Interface

Constructor Detail

Circle

```
public Circle(int diameter,
      int xPosition,
      int yPosition,
      java.lang.String color)
```

Create a new circle with x position, y position, diameter and color.

Method Detail

changeColor

```
public void changeColor(java.lang.String newColor)
```

Change the color. Valid colors are "red", "yellow", "blue", "green", "magenta" and "black".

Parameters:

`newColor` - The new color.

Loading class interface... Done.

saved

Class Edit Tools Options

Compile Undo Cut Copy Paste Find... Find Next Close Interface

changeSize

`public void changeSize(int newDiameter)`

Change the size to the new size (in pixels). Size must be >= 0.

Parameters:

`newDiameter` - The new diameter.

---

draw

`public void draw()`

Draw the circle with current specifications on screen.

---

erase

`public void erase()`

Erase the circle on screen.

Loading class interface... Done. *saved*

Class Edit Tools Options

Compile | Undo | Cut | Copy | Paste | Find... | Find Next | Close

Interface

moveHorizontal

```
public void moveHorizontal(int distance)
```

Move the circle horizontally by 'distance' pixels.

Parameters:

`distance` - The distance to move.

---

moveVertical

```
public void moveVertical(int distance)
```

Move the circle vertically by 'distance' pixels.

Parameters:

`distance` - The distance to move.

Loading class interface... Done.

saved

נכתוב תוכנית שמשתמשת במחלקה circle. אנו רוצים לצייר עיגול קטן בצבע צהוב ועיגול גדול יותר בצבע סגול.

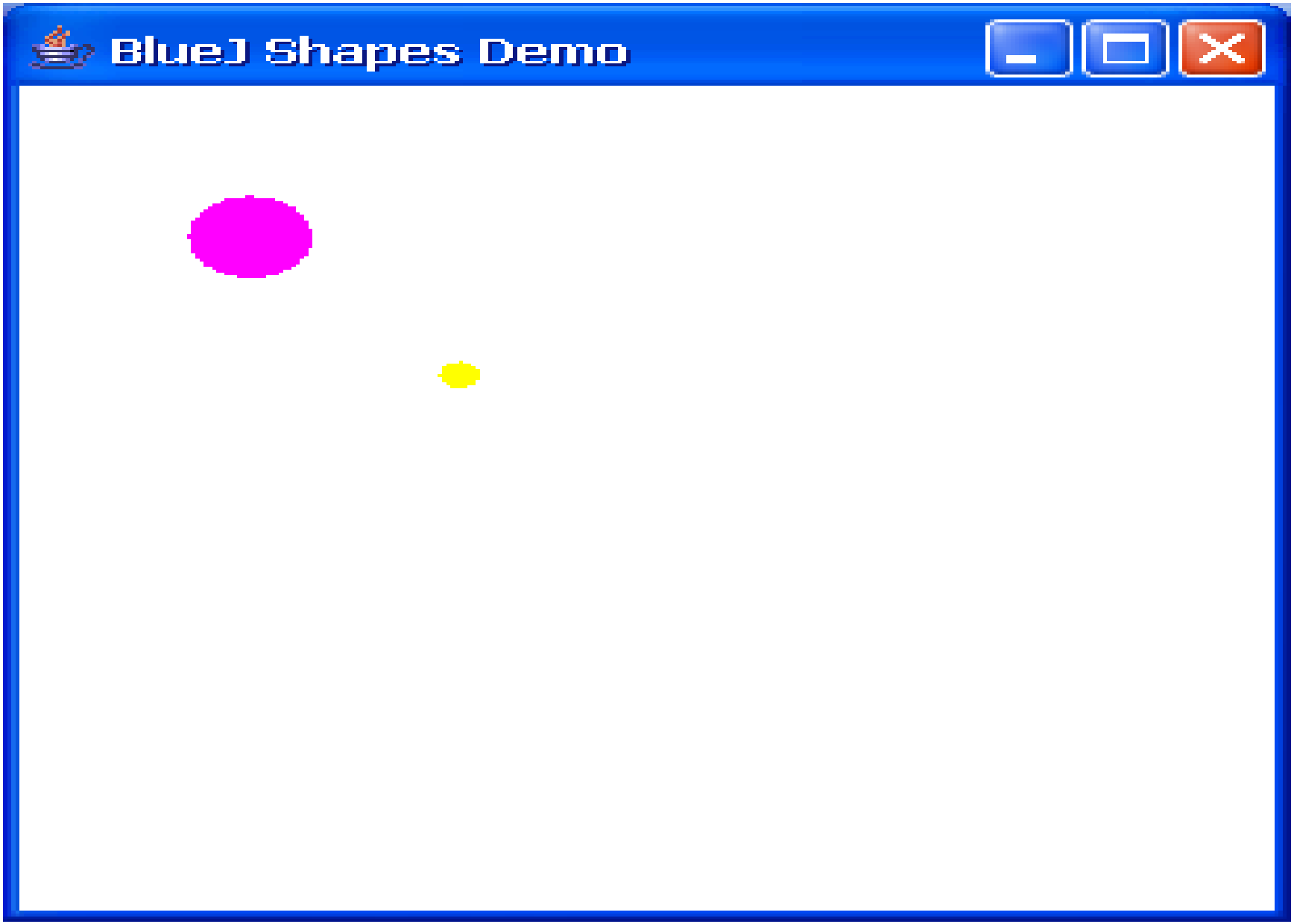
```
public class DrawCircles
{
    public static void main (String[]args){
        Circle smallCircle=new Circle(10,100,100,"yellow");
        smallCircle.draw();
        Circle bigCircle=new Circle(30,40,40,"magenta");
        bigCircle.draw();

}
}
```

קריאה לשיטה ()draw

הגדרת משתנה חדש מסוג circle ששמו smallcircle

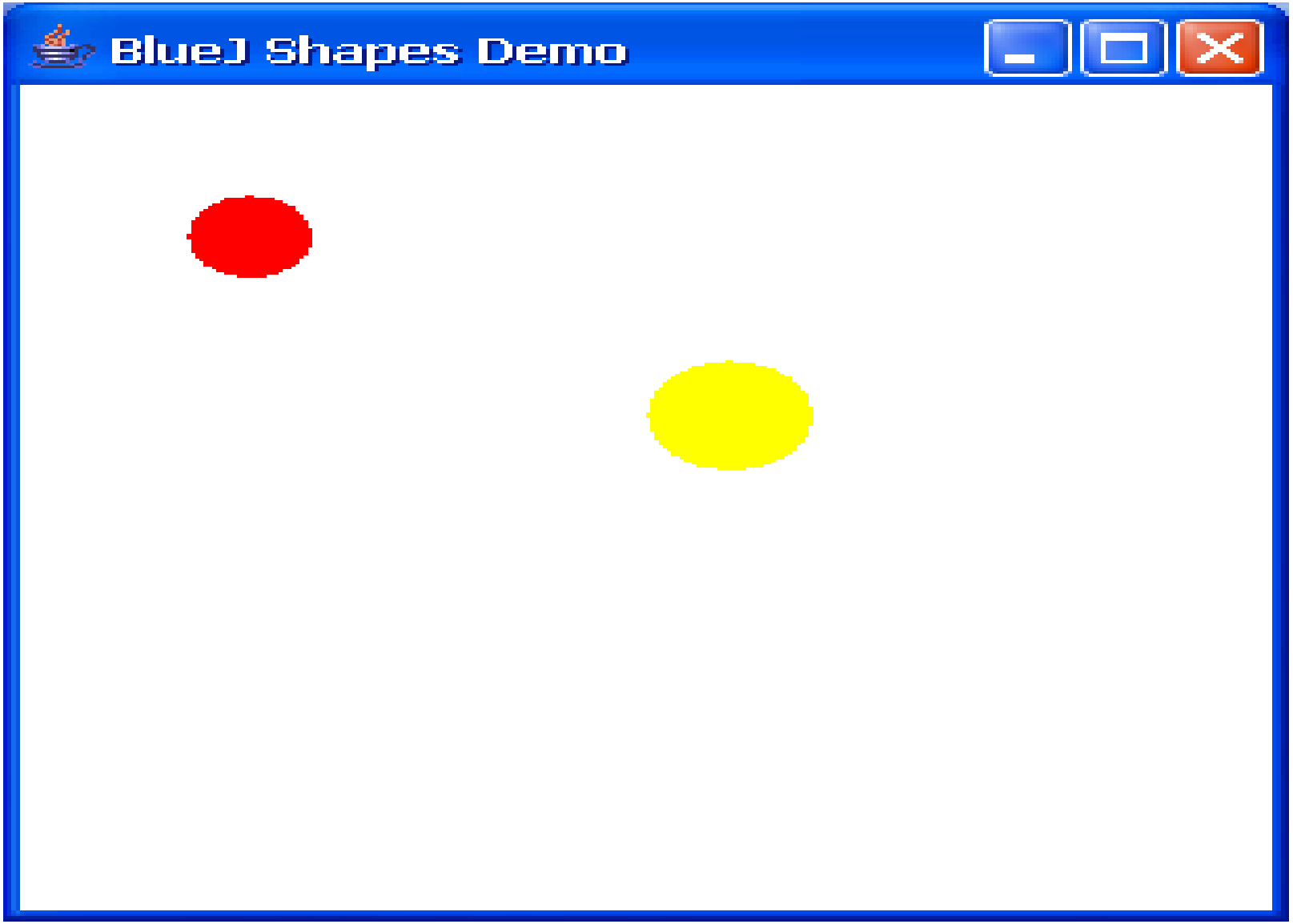
פקודה היוצרת אובייקט חדש בזכרון הערמה של המחשב

הפעלת הבנאי שיטה מיוחדת שמו כשם המחלקה אינה מחזירה ערך


BlueJ Shapes Demo

נבצע כמה שינויים

```
public class MakeChanges
{
   public static void main (String[]args){
        Circle smallCircle=new Circle(10,100,100,"yellow");
        Circle bigCircle=new Circle(30,40,40,"magenta");
        smallCircle.changeSize(40);
        bigCircle.changeColor("red");
        smallCircle.moveHorizontal(50);
        smallCircle.draw();
        bigCircle.draw();
}
}
```


BlueJ Shapes Demo

המחלקה Circle

```
import java.awt.*;
import java.awt.geom.*;
/**
 * A circle that can be manipulated and drawn or erased .
 * adapted from BlueJ example by Michael Kolling and David J. Barnes
 */
public class Circle
{
        private int _diameter;
        private int _xPosition;
        private int _yPosition;
        private String _color;
/**
 *     Create a new circle with x position, y position, diameter and color.
**/

    public Circle(int diameter,int xPosition,int yPosition,String color)
{
                _diameter = diameter;
                _xPosition = xPosition;
                _yPosition = yPosition;
                _color = color;// what really happened here?

}
```

הצהרות על תכונות האובייקטים

בנאי

```
/**
 *    Move the circle horizontally by 'distance' pixels.
*@     param distance.   The distance to move.
* /
    public void moveHorizontal(int distance)
{

            _xPosition += distance;

}

/**
 *    Move the circle vertically by 'distance' pixels.
*@     param distance.   The distance to move.
*/
    public void moveVertical(int distance)
{

            _yPosition += distance;

}
```

תעוד השיטות

שיטות המחלקה

```
/**
 *    Change the size to the new size (in pixels). Size must be >= 0.
*@    param newDiameter   The new diameter.
*/
   public void changeSize(int newDiameter)

{
          _diameter = newDiameter;
}
/**
 *    Change the color. Valid colors are "red", "yellow", "blue", "green",
*    " magenta" and "black".
*    @ param newColor   The new color.
*/
   public void changeColor(String newColor)
{
          _color = newColor;

}
```

חתימת השיטה

הפרמטרים שהשיטה מקבלת

שם השיטה

הערך המוחזר מהשיטה

מאפיין גישה

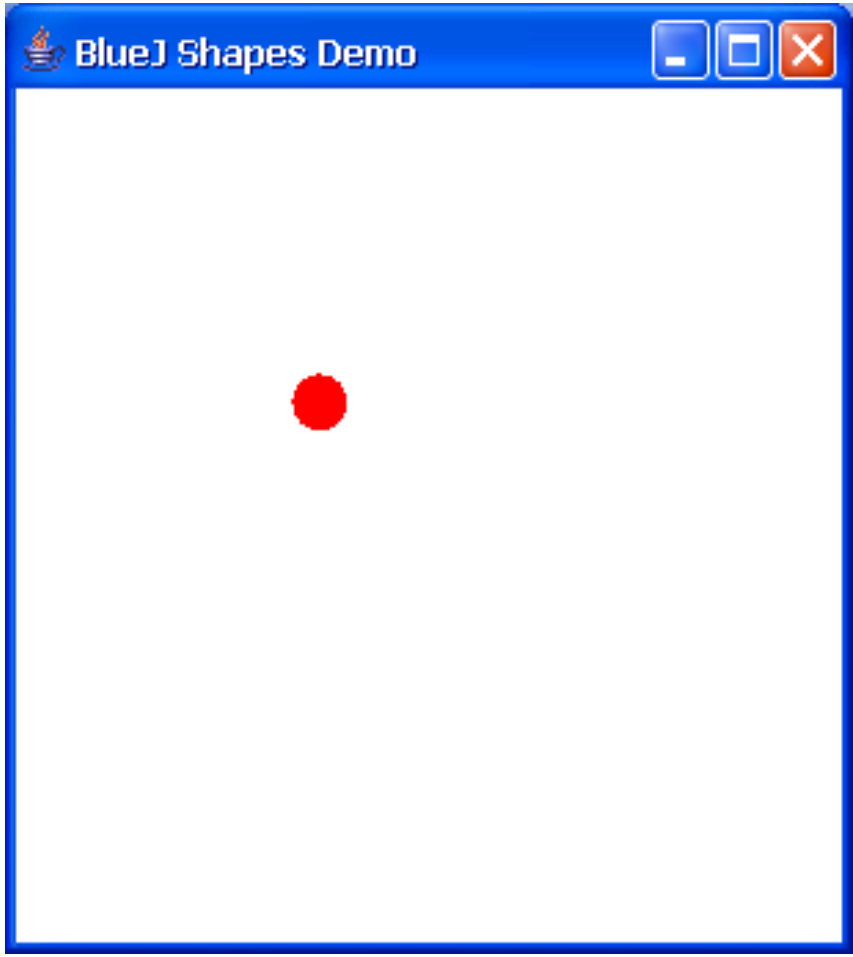
```
/**
 * Draw the circle with current specifications on screen.
*/
public void draw()        // we haven't learnt this
{

                    Canvas canvas = Canvas.getCanvas();
                    canvas.draw(this, _color, new Ellipse2D.Double(_xPosition, _yPosition,

                                                        _  diameter, _diameter));
                    canvas.wait(10);

}

/**
 * Erase the circle on screen.
*/
public void erase()
{

                    Canvas canvas = Canvas.getCanvas();
                    canvas.erase(this);

}
```

תעוד פנימי

נשווה בין התעוד במחלקה לבין מה שראינו בממשק

Aliasing

```
public class Aliasing
{
    public static void main (String[]args){
        Circle circle1=new Circle(20,100,100,"yellow");
        Circle circle2;
        circle2=circle1;
        circle2.changeColor("red");
        circle1.draw();
    }
}
```



רגע, circle1 היה בצבע צהוב, לא?

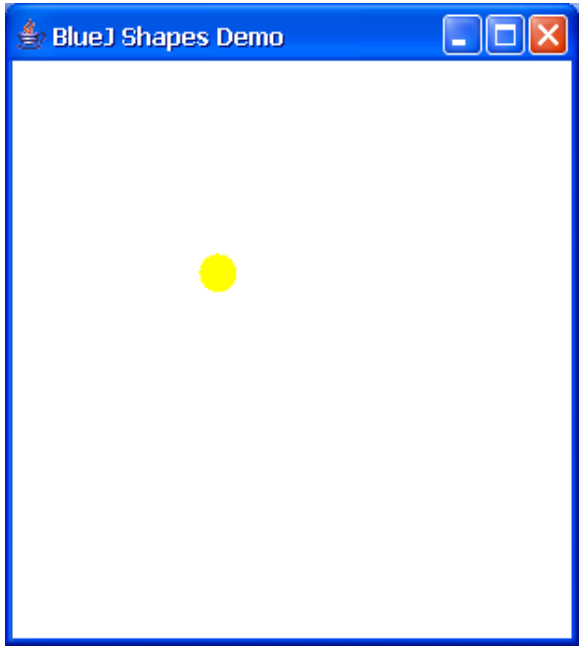
איך נוכל להעתיק אובייקט אחד לאובייקט אחר, ככה שיוצרו שני אובייקטים שונים בערמה?
לשם כך משתמשים בבנאי מיוחד שנקרא בנאי העתקה.

בנאי העתקה במחלקה Circle

```
/**
 *     Create a new circle identical to a given circle.
**/
public Circle(Circle c)
{
        _diameter = c._diameter;
        _xPosition = c._xPosition;
        _yPosition = c._yPosition;
        _color = c._color; // what really happened here?

}
```

Copy Constructor

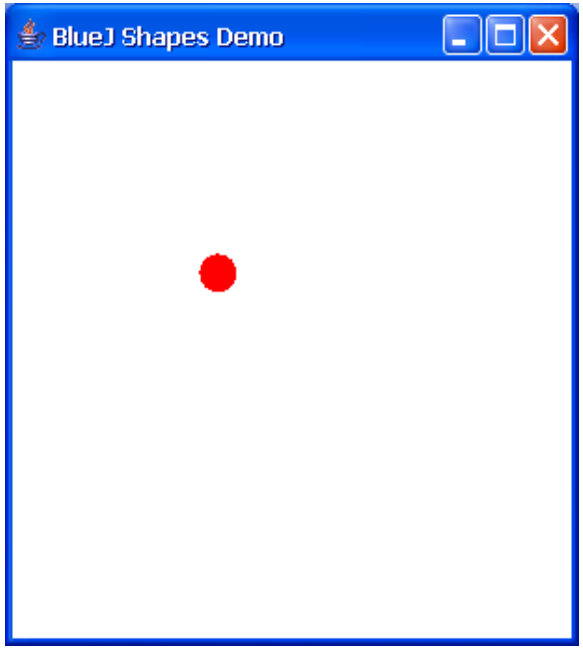
```
public class CopyConstructor
{
    public static void main (String[]args){
        Circle circle1=new Circle(20,100,100,"yellow");
        Circle circle2 = new Circle(circle1);
        circle2.changeColor("red");
        circle1.draw();
    }
}
```



circle1 נשאר בצבע צהוב, ומה עם circle2?

Copy Constructor

```
public class CopyConstructor
{
    public static void main (String[]args){
        Circle circle1=new Circle(20,100,100,"yellow");
        Circle circle2 = new Circle(circle1);
        circle2.changeColor("red");
        circle2.draw();
    }
}
```



מה היינו צריכים לעשות עם הצבע?

```
 public Circle(int diameter,int xPosition,int yPosition,String color)
{
        _diameter = diameter;
        _xPosition = xPosition;
        _yPosition = yPosition;
        _color = new String(color);// copy constructor of String

}
  public Circle(Circle c)  // copy constructor of Circle
{
        _diameter = c._diameter;
        _xPosition = c._xPosition;
        _yPosition = c._yPosition;
        _color = new String(c._color);// copy constructor of String

}
```

גם ל-String יש בנאי העתקה!

האם נחוץ ב-2 המקרים לקרוא לבנאי ההעתקה של string?

נשתמש בשיטת equals כדי לראות אם 2 מעגלים שווים

```
/**
*Compare current circle with another circle. If same return true otherwise return false.
*@param c.   The circle to be compared to.
*@return   true if both circles are the same, false if different.
*/
public boolean equals (Circle c)
{
if( ( _diameter == c._diameter)&&
     (_xPosition == c._xPosition)&&
     (_yPosition ==c._yPosition)&&
     (_color.equals( c._color)))
          return (true);
 else
        return (false);
}
}
```

```
public class Equals
{
    public static void main (String[]args){
        Circle circle1=new Circle(20,100,100,"yellow");
        Circle circle2 = new Circle(20,100,100,"yellow");
        if(circle1.equals(circle2)) //==true
            System.out.println("Both circles are identical");
        else
            System.out.println("The circles are different");
       circle2.changeColor("red");
        if(circle1.equals(circle2)) //==true
            System.out.println("Both circles are identical");
        else
            System.out.println("The circles are different");
    }
}
```

הפלט:

Both circles are identical

The circles are different

toString

המטרה: ליצור ייצוג של אובייקט כמחרוזת, כדי שנוכל להדפיס תכונות של אובייקט כלשהו.

השיטה toString

```
/**
 *     return  circle properties into a string
*@     return   return the string of circle properties
* /
public String toString()
{
return "x=  " + _xPosition + " y=  " + _yPosition+ " diameter=  "+ _diameter+ " color= " +_color;
}
```

הדפסת המעגלים

```
public class PrintCircles
{
    public static void main (String[]args){
        Circle smallCircle=new Circle(10,100,100,"yellow");
        System.out.println (smallCircle);
    //smallCircle.toString()
        Circle bigCircle=new Circle(30,40,40,"magenta");
        System.out.println (bigCircle); //bigCircle.toString()

}
}
```

הפלט:

```
x=  100 y=  100 diameter=  10 color= yellow
x=  40 y=  40 diameter=  30 color= magenta
```

פקודת ההדפסה מקבלת כפרמטר אובייקט מטיפוס String. כאשר הקומפיילר מחכה לאובייקט מטיפוס String ואין לו כזה, הוא אוטומטית קורא לשיטה toString המוגדרת במחלקה.

Encapsulation

- תכונה חשובה מאד של OOP
- מחלקה תחשוף כלפי חוץ רק את הממשק שנועד לשימוש המשתמש
- המנגנונים הפנימיים מוסתרים
- פועל כקופסה שחורה
- בדוגמה של המחקלה Circle, אין למשתמש גישה לתכונות הפנימיות של המעגל. הוא יכול לבצע שינויים רק בעזרת השיטות.

Access modifiers

בעזרת מאפייני הגישה , נוכל לממש כימוס

public

יש גישה לשיטות שנמצאות מחוץ למחלקה

public void draw()

יש גישה לשיטת draw גם מחוץ למחלקה Circle. כיצד היינו מציירים מעגל אילו השיטה היתה מוגדרת כ-private?

private

יש גישה רק לשיטות של המחלקה

```
private int _diameter;
private int _xPosition;
private int _yPosition;
private String _color;
```

רק שיטות של המחלקה Circle יכולות לגשת או לשנות את התכונות של המעגלים

ואם בכל זאת נרצה לאפשר גישה לתכונות הפנימיות מחוץ למחלקה

הפתרון המקובל הוא לספק שיטות ציבוריות (public) שדואגות לנהל את האינטראקציה עם התכונות הפרטיות, כך שנוכל לשנות את התכונות או לקבל מידע על התכונות.

לדוגמה

```
public void setXPosition(int xPosition)
  {
   _xPosition= xPosition;
}
public int getXPosition()
{
  return _xPosition;
}
```

Mutator method

Accessor method

הבנאי הריק

כאשר ציירנו עיגולים ב-BlueJ, היתה ברירת מחדל למצב ההתחלתי של המעגל. אנחנו בקשנו מעגל חדש, וקבלנו מעגל עם תכונות קיימות.

נוסיף בנאי ריק גם למחלקה שלנו.

```
/**
 *     Create a new circle at default position, size and  color.
**/



    public Circle()
{
        _diameter = 30;
        _xPosition = 20;
        _yPosition = 60;
        _color = "blue";

}
```

ואם נרצה להוסיף בנאי נוסף שמקבל רק את מיקום המעגל וקובע כברירת מחדל גודל וצבע?

```
/**
 *     Create a new circle at default position.
**/



   public Circle(int xPosition, int yPosition)
{
            _xPosition = xPosition;
            _yPosition = yposition;
            _diameter = 30;
            _color = "blue";

}
```

בנאים 4

Circle

Class Edit Tools Options

Compile Undo Cut Copy Paste Find... Find Next Close Interface

Constructor Summary

Circle() Create a new circle at default position with default color.
Circle(Circle c) Create a new circle identical to a given circle.
Circle(int xPosition, int yPosition) Create a new circle at default position.
Circle(int diameter, int xPosition, int yPosition, java.lang.String color) Create a new circle with x position, y position, diameter and color.

Loading class interface... Done.

saved

```
  public Circle()
{
        _diameter = 30;
        _xPosition = 20;
        _yPosition = 60;
        _color = "blue";

}
public Circle(int diameter,int xPosition,int yPosition,String color)
{
        _diameter = diameter;
        _xPosition = xPosition;
        _yPosition = yPosition;
        _color = new String(color);// copy constructor of String


}

public Circle(int xPosition, int yPosition)
{
        _xPosition = xPosition;
        _yPosition = yPosition;
        _diameter = 30;
        _color = "blue";

}

public Circle(Circle c)
{
        _diameter = c._diameter;
        _xPosition = c._xPosition;
        _yPosition = c._yPosition;
        _color = new String(c._color);// copy constructor of String
}
```

בנאי ריק

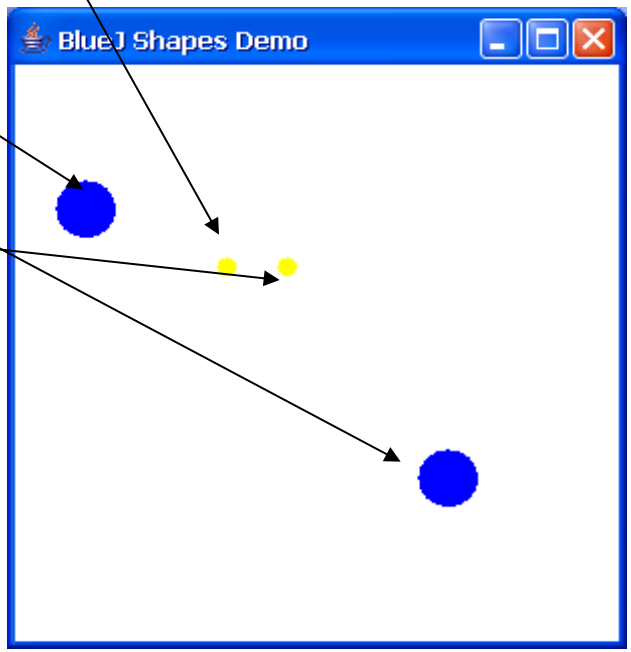
בנאי שמקבל את כל התכונות

בנאי שמקבל רק מיקום

בנאי העתקה

שימוש בבנאים השונים

```
public class UseConstructors
{
    public static void main (String[]args){
        Circle smallCircle=new Circle(10,100,100,"yellow");
        smallCircle.draw();
        Circle defaultCircle=new Circle();
        defaultCircle.draw();
        Circle downCircle=new Circle(200,200);
        downCircle.draw();
        Circle likeSmallCircle=new Circle(smallCircle);
        likeSmallCircle.moveHorizontal(30);
        likeSmallCircle.draw();
}
}
```



2 שיטות נוספות

מדוע private?

```
private void drawX(Circle c)
 {

  c.draw();
   Circle c5=new Circle (c._diameter,c._xPosition+6+c._diameter,c._yPosition+6+c._diameter,c._color);
  c5.draw();
  Circle c6=new Circle (c._diameter,c._xPosition+6+c._diameter,c._yPosition-6-c._diameter,c._color);
  c6.draw();
  Circle c7=new Circle (c._diameter,c._xPosition-6-c._diameter,c._yPosition+6+c._diameter,c._color);
  c7.draw();
  Circle c8=new Circle (c._diameter,c._xPosition-6-c._diameter,c._yPosition-6-c._diameter,c._color);
  c8.draw();

   }
```

2 שיטות נוספות

```
private void drawFlower(Circle c)
 {
    c.draw();
    Circle c1=new Circle
     (c._diameter,c._xPosition+c._diameter+10,c._yPosition,c._color);
    c1.draw();
    Circle c2=new Circle (c._diameter,c._xPosition-c._diameter-10,c._yPosition,c._color);
    c2.draw();
    Circle c3=new Circle
     (c._diameter,c._xPosition,c._yPosition+10+c._diameter,c._color);
    c3.draw();
    Circle c4=new Circle (c._diameter,c._xPosition,c._yPosition-10-c._diameter,c._color);
    c4.draw();
    drawX(c);
      }
```

שיטה נוספת

chooseWhatToDraw

השיטה מקבלת עיגול נוסף ותו: O או X.

אם התו הוא O השיטה שולחת את העיגול שקבל את ההודעה לשיטת drawFlower, אם התו הוא X, השיטה שולחת את העגול שהתקבל כפרמטר לשיטת drawX.

מה שם האובייקט שנשלח?

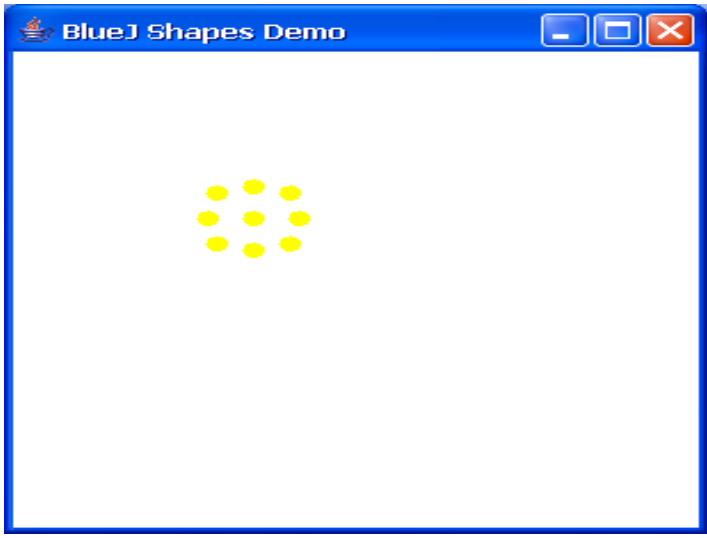
this

```
/**
 * choose whether to draw a flower around the object which received
* the message or an X around the object sent as a parameter
 */
 public void chooseWhatToDraw(char c, Circle circ)
 {
    if(c=='O' )
         drawFlower(this);
     if(c=='X')
          drawX(circ);

  }
```

האובייקט שקבל את ההודעה

האובייקט שהתקבל כפרמטר

```
public class This
{
  public static void main (String[]args){
    Circle circle1=new Circle(10,100,100,"yellow");
    Circle circle2=new Circle(10,100,100,"magenta");
    circle1.chooseWhatToDraw('O', circle2);
}
}
```



```
public class This
{
  public static void main (String[]args){
      Circle circle1=new Circle(10,100,100,"yellow");
      Circle circle2=new Circle(10,100,100,"magenta");
     circle1.chooseWhatToDraw('X', circle2);
}
}
```