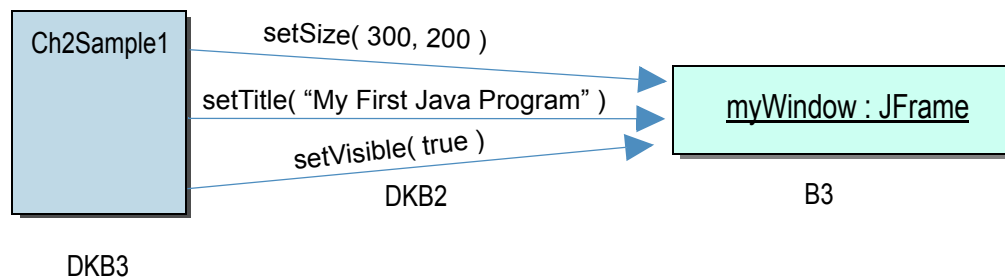


Diagram Color Scheme

Please use the following color scheme consistently throughout the whole book. Here are the rules with examples from Ch 1 to Ch 4.

Rules

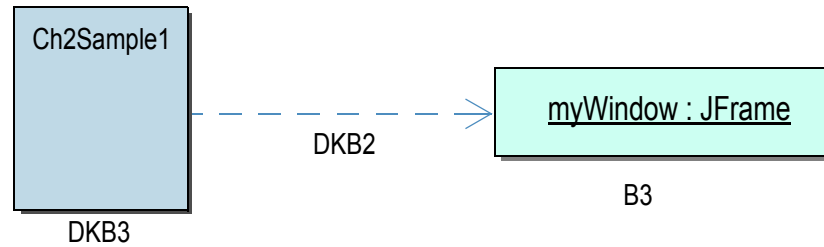
1. Class in DKB3. A class is drawn as a rectangle and its name is in the rectangle.
2. Object (instance) in B3. An object is drawn as a rectangle and its name is always underlined and has a colon.
3. Messages is drawn as a solid line in DKB2. The arrow is a filled arrow.



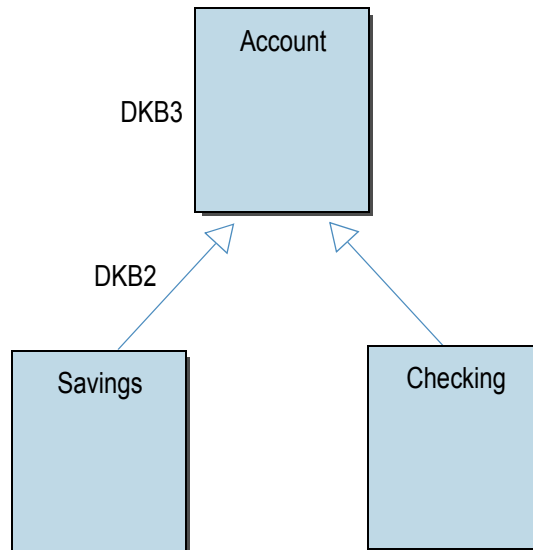
2

Diagram Color Scheme

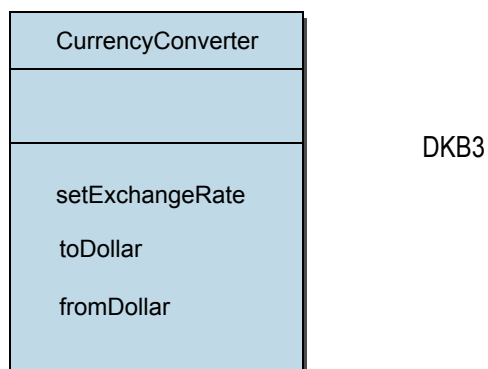
4. Dependency relationship is drawn as a dashed line in DKB2. An arrow is drawn as two short lines.



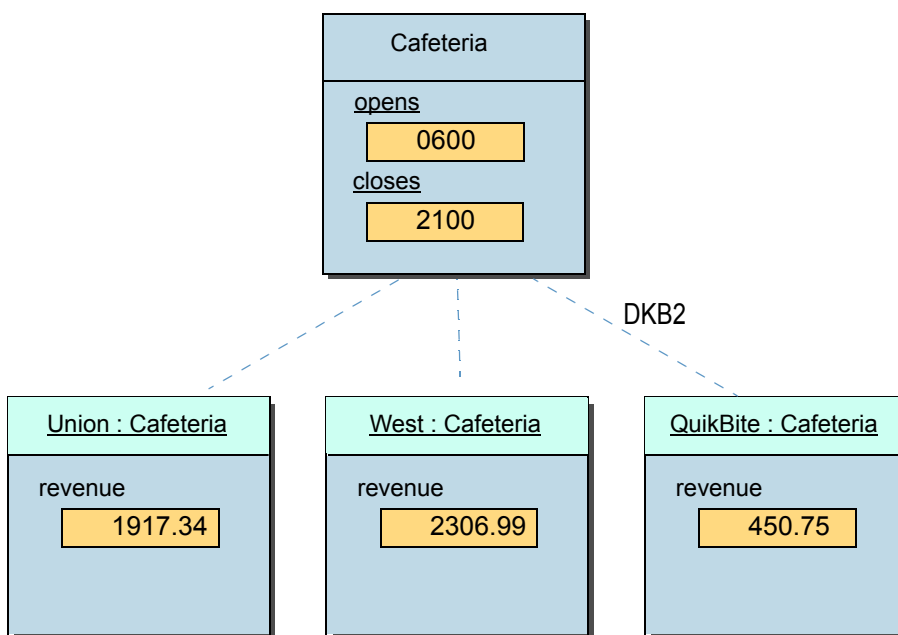
5. Inheritance in a hollowed arrow in DKB2. Do not merge; draw each inheritance line separately.



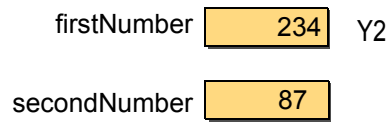
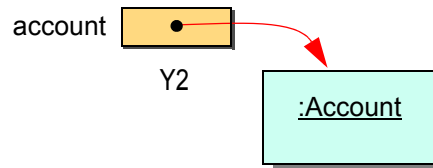
6. Class may be drawn with data member and method components. All parts are drawn in the same DKB3.



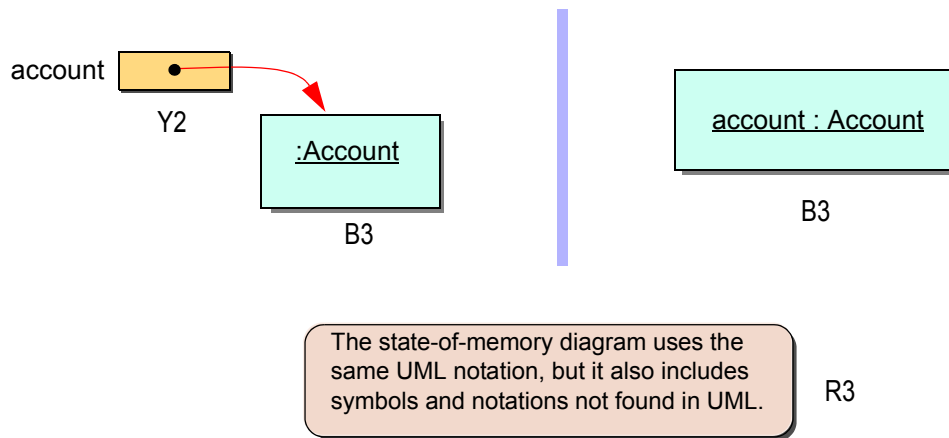
7. Instance-of relationship in dashed line in DKB2. Do not merge, draw lines separately.



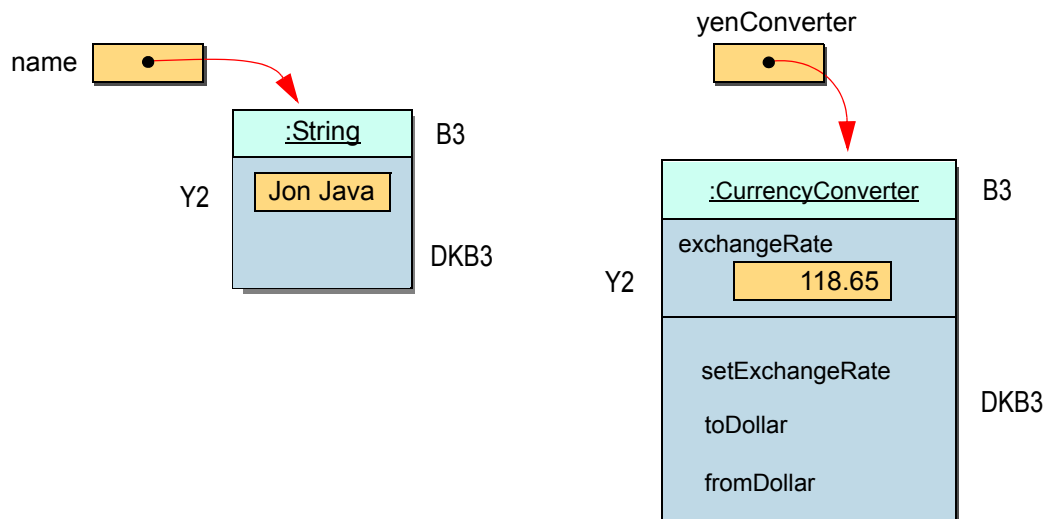
8. Variable box that contains a primitive data value or an address (red arrow) is in Y2.



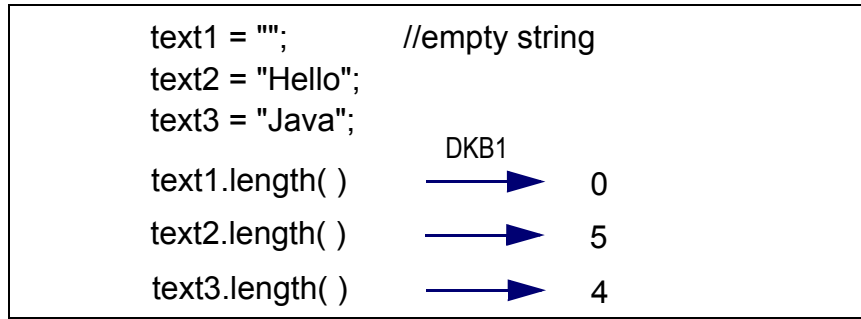
9. Explanation box in R3. It is a rounded rectangle containing explanation in the diagram.



10. When an object is drawn with data member shown, then the variable box is in Y2, data member area (bottom part) in DKB3 (class color) and object name part (underlined text with colon) in B3 (object color).



11. Arrows that shows the result of some operations or assignments in DKB1.



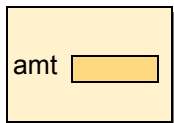
12. For the state-of-memory diagram, the memory box that contains variable box (Y2) is in Y3.

13. An arrow for execution flow in TX3.

1

```
amt = yenConverter.fromDollar( 200 );
```

at **1** before calling **fromDollar**



state of memory

1

execution flow

TX3

```
public double fromDollar( double dollar ) {
    double amount, fee;

    fee      = exchangeRate - feeRate;
    amount = dollar * fee;

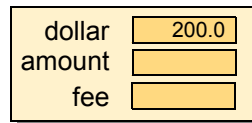
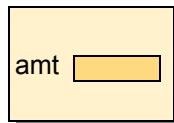
    return amount;
}
```

Local variables do not exist before the method execution.

2

```
amt = yenConverter.fromDollar( 200 );
```

at **2** after declaration

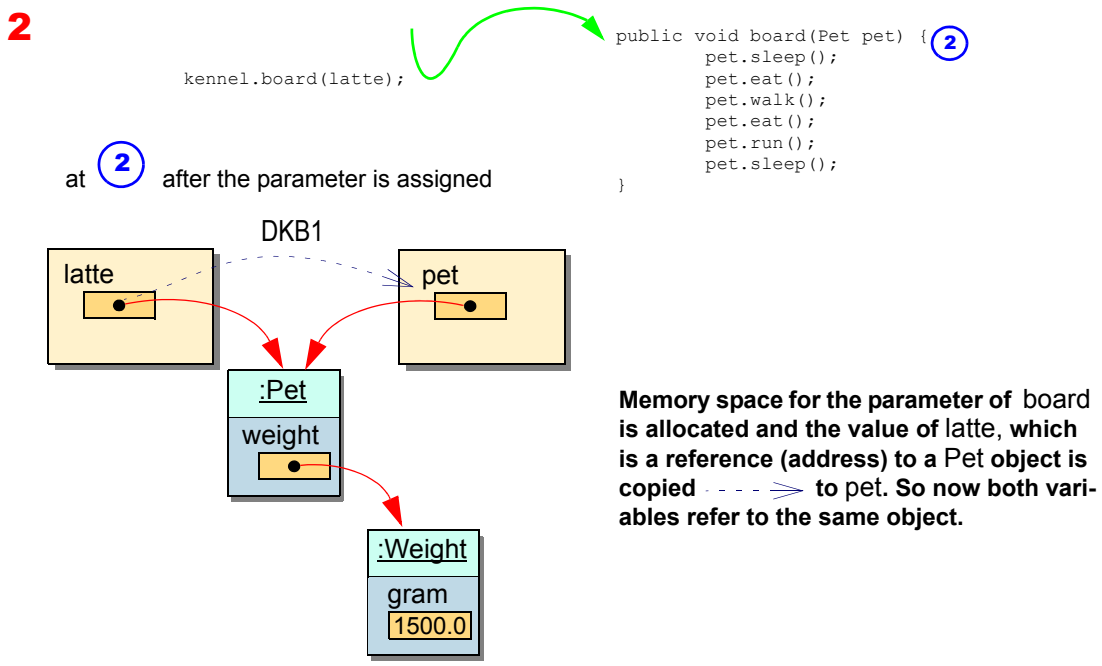
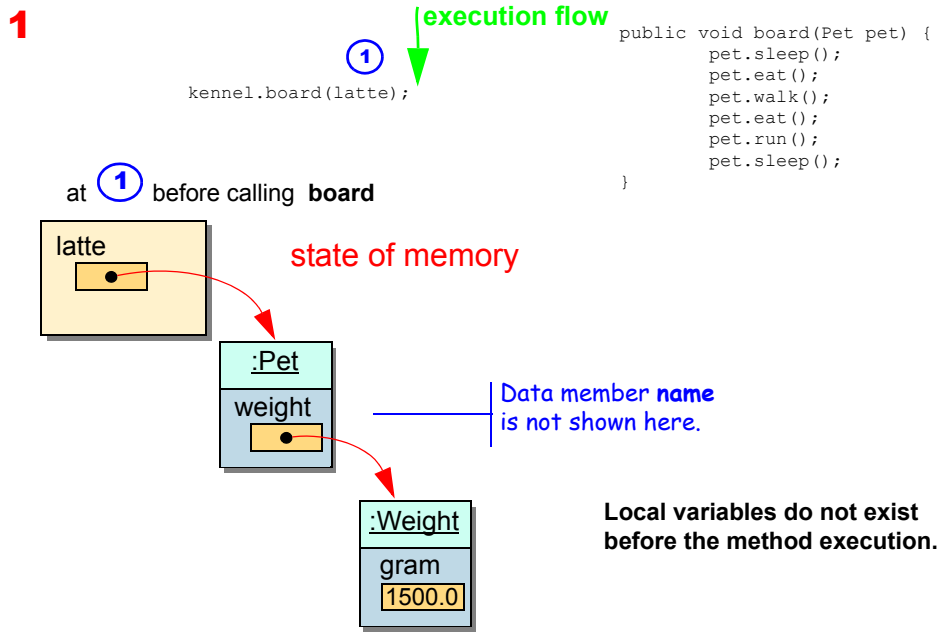


```
public double fromDollar( double dollar ) {
    double amount, fee; 2
    fee      = exchangeRate - feeRate;
    amount = dollar * fee;

    return amount;
}
```

Memory space for the local variables and parameter of fromDollar is allocated.

14. A dashed line for passing of a value in DKB1.



15. Reserved words in bold DKB2. String literals, surrounded in double quotes, in plain DKB2. (I wanted to use blue, but is not available in the color palette. DKB1 is too dark and can't tell the difference from black). Comments in TX3. Parentheses, brackets and braces in bold red (TX2?).

```

/*
  Introduction to OOP with Java 3rd Ed, McGraw-Hill

  Chapter 6 Sample Program: Dropping a Watermelon

  File: Ch6DroppingWaterMelon.java
*/

import javabook.*;
import java.io.*;

class Ch6DroppingWaterMelon {

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

        double  initialHeight,
                position,
                touchTime;

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

        System.out.print("Initial Height:");
        initialHeight = Double.parseDouble(bufReader.readLine());

        touchTime      = Math.sqrt(initialHeight / 16.0);
        touchTime      = Math.round(touchTime * 10000.0) / 10000.0;
                        //convert to four decimal places

        System.out.println("\n\n   Time t       Position at Time t \n");

        for (int time = 0; time < touchTime; time++) {
            position = -16.0 * time*time + initialHeight;
            System.out.print("   " + time);
            System.out.println("       " + position);
        }

        //print the last second
        System.out.println("   " + touchTime + "       0.00");

        System.out.println("\n\n\n");
    }
}

```