UNIVERSITY OF TWENTE.

APPLICATION DEVELOPMENT

LECTURE 3: DESIGN A CLASS, USING OBJECTS AND METHODS, CONDITIONS AND LOOPS

```
class AppDev {
}
```

Part of SmartProducts

INTRODUCTION

APPLICATION DEVELOPMENT

Fjodor van Slooten
W241 *(Horst-wing West)*
f.vanslooten@utwente.nl

- Design a class
- Using objects and methods
- Conditions and loops
- Assignment

slides @ vanslooten.com/appdev

ASSIGNMENT 2

- Adding a variable and a method
- A method declaration (definition) and it's use (call)

```
JButton btnDraw = new JButton("Draw");
btnDraw.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        String r = textFieldR.getText();
        System.out.println("Input value for red: "+r);
        String g = textFieldG.getText();
        System.out.println("Input value for green: "+g);
        String b = textFieldB.getText();
        System.out.println("Input value for blue: "+b);

        // prevent errors:
        if (!r.matches("\\d+")) { r="0"; textFieldR.setText(r); }
        if (!g.matches("\\d+")) { g="0"; textFieldG.setText(g); }
        if (!b.matches("\\d+")) { b="0"; textFieldB.setText(b); }

        // get integer-value from String r:
        int rValue = Integer.parseInt(r);
        int gValue = Integer.parseInt(g);
        int bValue = Integer.parseInt(b);

        // call method setColor() of panelDraw:
        panelDraw.setColor(rValue, gValue, bValue);
    }
});
```

```
public class DrawingPanel extends JPanel {

  Color drawColor = Color.yellow;

  protected void paintComponent(Graphics g) {
    super.paintComponent(g);
          ...
  }

  public void setColor(int r, int g, int b) {
    drawColor = new Color(r % 256, g % 256, b % 256);
    repaint(); // draw again because the color has been changed.
  }
```

Declaration

Use (call)

Find declaration? Select and press F3 (or right-click)

DESIGN A CLASS

ANALYZE OBJECT (IN REAL WORLD)

Properties

Position (x,y)
Diameter
Color

Ball

Actions/behavior

Move
Bounce (change direction)
Draw

(methods)

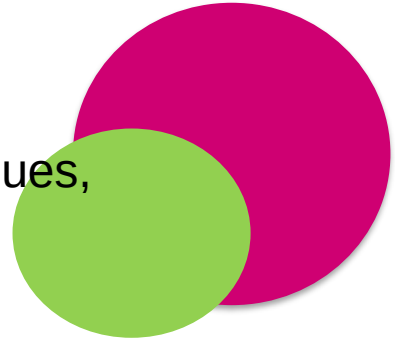
What (is)? → Properties
What can … do? → Methods
How can … do? → (Pseudo) code (of methods)

DESIGN A CLASS

DETAIL CLASS IN (PSEUDO) CODE

Pseudo code: incomplete code, human-readable

- Types of properties
- Methods: return values, parameters



```
public class Ball {
  // properties:
  int x, y; // position
  int diameter;
  Color color;

  // methods:
  public void move();
  public void bounce();
  public void draw(Graphics g);
}
```

Standard return value: **`void`** (=nothing), and modifier **`public`**

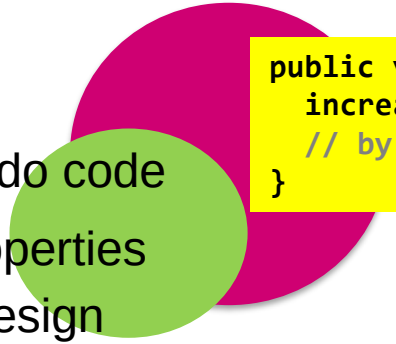
Parameter **`g`**, of type **`Graphics`**, needed for drawing

DESIGN A CLASS

DETAIL METHODS IN (PSEUDO) CODE

Pseudo code: incomplete code, human-readable

- For each method:
- Write steps in pseudo code
- If new variables/properties are needed, alter design



```
public void move() {
  increase position (x,y)
  // by what? introduce dx/dy? (delta x and y)
}
```

```
public void bounce() {
  reverse direction:
  dx = -dx
  dy = -dy
}
```

```
public void draw(Graphics g) {
  set color
  draw filled circle at position (x,y)
}
```

Next step: start coding

CREATE BALLS: NEW

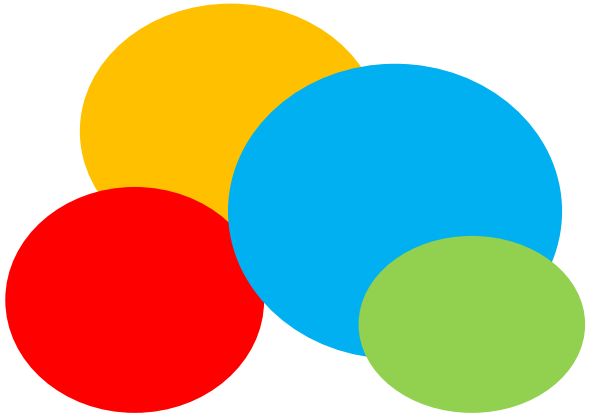
Head First: p240-249, **Aan de slag met:** 2.5.3, 4.10

b1 is a new object of type **Ball:**
"b1 is a Ball"

```
Ball b1 = new Ball(10, Color.orange, 10, 20);
Ball b2 = new Ball(8,  Color.red,     5, 30);
Ball b3 = new Ball(15, Color.blue,   20, 25);
Ball b4 = new Ball(5,  Color.green,  30, 30);
```

Call to constructor

Parameters determine difference in properties



Properties (class variables) get a value

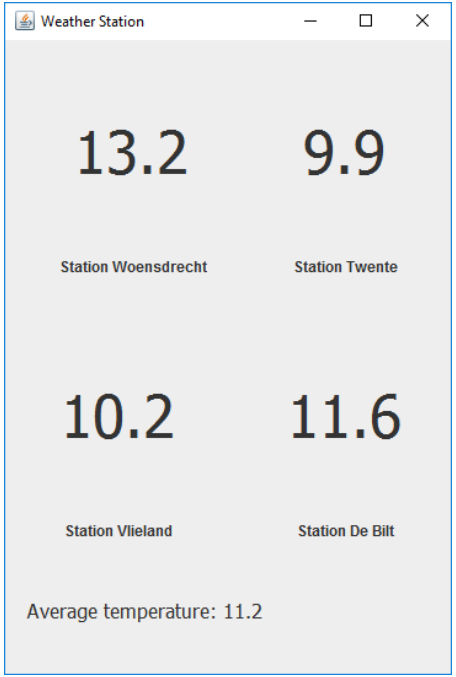
```
// constructor assigns properties:
public Ball(int d, Color c, int i, int j) {
  diameter = d;
  color = c;
  x = i;
  y = j;
}
```

Constructor: special method with same name as class and no return value

USING OBJECTS

- Assignment: weather-panels, create a class once, use 4 times

```
panel = new TemperaturePanel(6340);

panel2 = new TemperaturePanel(6290);

panel3 = new TemperaturePanel(6260);

panel4 = new TemperaturePanel(6235);
```

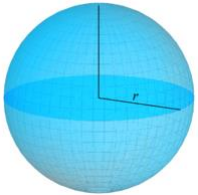


METHODS: RETURN VALUE

Surface area [edit]

The surface area of a sphere is:

$$A = 4\pi r^2.$$



en.wikipedia.org/wiki/Sphere

Type of result: **double**

Parameter

```
public double calculateSurfaceArea(double r) {
  double A;
  A = 4 * Math.PI * Math.pow(r,2);
  return A;
}
```

Return the value (to caller)

Calculation

Use the method:

Parameter passed to method

```
double result = calculateSurfaceArea(10);
```

Call of the method

More info

‘CALL’ A METHOD
WITH TEXT AS A PARAMETER

Head First: p74-76 ***Aan de slag met*:** 4.9 & 4.10

Variable **temp** gets a value

‘Call’ method **readTemperature()** of object ***w***

“Dear weather station, please read the temperature for us”

```
String temp = w.readTemperature();

labelTemp.setText(temp);
```

“Show temperature in userinterface”

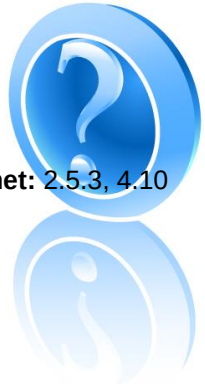
Call method **setText** to show String temp in a label

Variable **temp** is used as a parameter in a method-call

CONDITIONS

IF …

Head First: p10-13 **Aan de slag met:** 2.5.3, 4.10



Condition between (…)

```
int x = 3;
if (x == 3) {
   System.out.println("x must be 3");
}
```

(Conditional or Boolean) Operators

<	smaller?
<=	smaller or equal?
>	larger?
>=	larger or equal?
==	equal?
!=	not equal?

x=5	x gets value 5 (assignment)
x==5	does x equal 5 ?

CONDITIONS

IF … ELSE …

```
int age = 14;
int length = 110;

if (age < 10 && length > 110)
  System.out.println("You are a tall kid");
else if (age > 10 && length <= 110)
  System.out.println("Eat more bananas!");
else
  System.out.println("I guess you are Ok");
```

Logical operators

&&	and
||	or
!	not

Use to build boolean expressions.
Result is *true* or *false*.

More info

CONDITIONS

SWITCH

```
switch(x) {
  case 1:
          soundbite = new File("cat.wav"); break;
  case 2:
          soundbite = new File("chicken.wav"); break;
  case 3:
          soundbite = new File("cow.wav"); break;
  case 4:
          soundbite = new File("dog.wav"); break;
  case 5:
          soundbite = new File("frog.wav"); break;
  default:
          soundbite = new File("bird.wav");
}
```

default: if none of the options complies

More info

REPEAT: LOOPS

WHILE …

Head First: p5,10-13 **Aan de slag met:** 5.8-5.14

Condition between (…)

```
Dog rufus = new Dog();

int x = 0;
while (x < 3) {
  rufus.bark();
  x = x + 1;
}
```

How many times does rufus bark?

Condition depends on x!

More info

REPEAT: LOOPS

FOR …

Control variable | Condition | Step for control variable

```
for (int l=0; l<4; l++) {
  System.out.println("Line "+l);
}
```

What is the output of these loops?

```
for (int l=0; l<8; l++) {
  for (int c=0; c<l; c++)
        System.out.print("#");
  System.out.println("");
}
```

increase a variable x by one: x++
same as: x = x + 1

More info

FORMAT OUTPUT

STRING.FORMAT

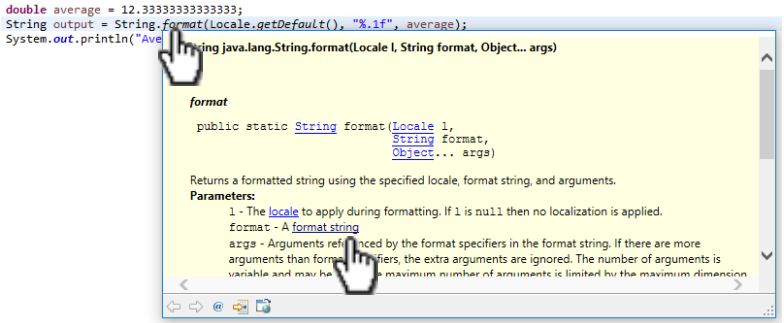
Head First: p10-13 **Aan de slag met:** 2.5.3, 4.10

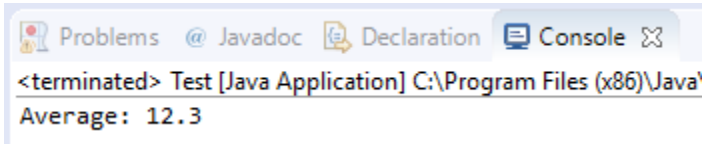
Format according to system's default locale settings

```
double average = 12.33333333333333;

String output = String.format(Locale.getDefault(), "%.1f", average);

System.out.println("Average: "+output);
```

Format as a decimal number with one digit after the . (.1f)

Input parameter(s)





SCOPE OF VARIABLES

{ Scope: region in code where a variable (or object) is valid. }

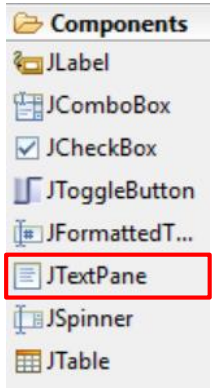
```
public class TemperaturePanel extends JPanel {
  WeatherStation w;

  public TemperaturePanel(int id) {
      JLabel labelTemp = new JLabel("25.7");

      w = new WeatherStation(id);

      String temp = w.readTemperature();

      labelTemp.setText(temp);
  }

  public double getTemp() {
      String temp = w.readTemperature();
      ...
  }
}
```

Object **w** is a *class-variable* in class **TemperaturePanel**

temp is a *local variable* valid in the constructor

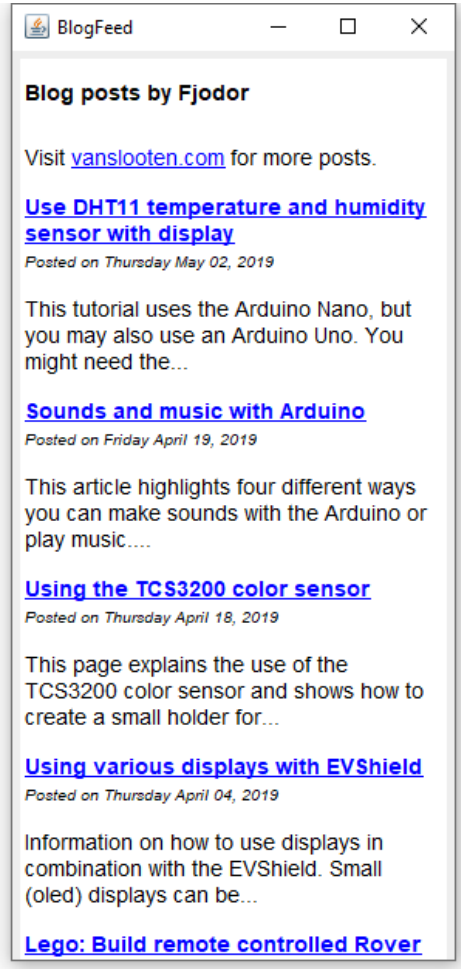
Object **w** can be used by all methods

LIBRARIES



- **`java.net, org.w3c, javax.xml`**: Libraries for internet applications & XML
- XML: Extensible Markup Language (Standard Data Exchange)
- Show webpage in textPane:

```
try {
   textPane.setPage("https://home.et.utwente.nl/slootenvanf/feed.php");
} catch (IOException e) {
   e.printStackTrace();
}
```



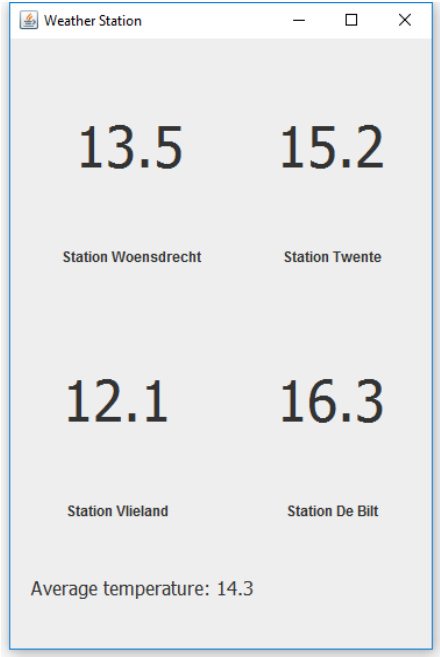
ASSIGNMENT #3

Deadline of each assignment is the next session:
so you can have this assignment checked no later than the next lecture

- “Create an application that can show weather-data from multiple weather stations”
- Have your work checked! (at table in front of room)

  Checkpoint

- Extra challenge & appendix: get temperature from connected Arduino
- Try examples/self-study



Slides, assignments etc @ vanslooten.com/appdev