



Java – Solution of assignments



Paolo Vercesi
Technical Program Manager

Assignment

Implement a *Calculator* class to perform arithmetic operations.

```
$ java Calculator 6 + 4.1
10.1
$ java Calculator 3.6 / -2
-1.8
$ java Calculator 8.5 * 9
76.5
$ java Calculator -3.14
-3.14
```

I let you discover how to convert strings to numbers

Enhance the calculator so that it can handle concatenated operations

```
$ java Calculator 6 + 4.1 * 3
10.1
30.3
$ java Calculator 3.6 / 2 + -0.3 / .5
1.8
1.5
3
```



“Problem decomposition”

What am I asked to do?

- 1. run a program using the java command*
- 2. take input from the command line*
- 3. interpret the arguments on the command line and execute the indicated operation*

Do I know how to do the actions in the points 1, 2, and 3?

Are there unknowns?



Resolving unknowns with the “scientific method”

Analyze your problem

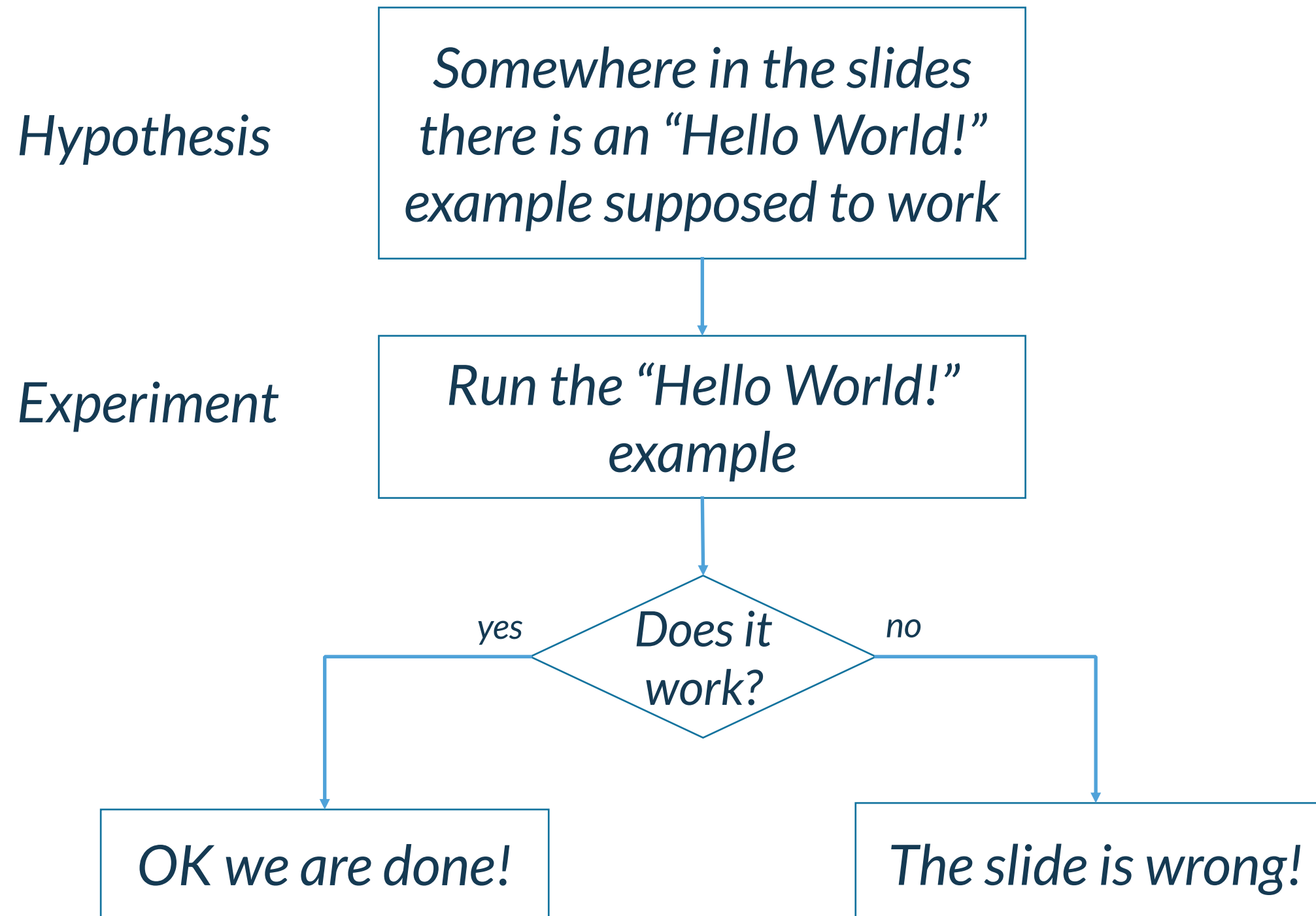
Formulate a hypothesis

Test the hypothesis

Observe the results and draw your conclusion



1. Run a program using the java command



2. Take input from the command line

Hypothesis

*we can use the command line argument
contained in `String[] args`*

Experiment

PrintCommandLineArgs.java

```
public class PrintCommandLineArgs {  
  
    public static void main(String[] args) {  
        for (String arg : args) {  
            System.out.println(arg);  
        }  
    }  
}
```

Results

```
$ java Calculator 3.14 + 15  
3.14  
+  
15
```



3. Interpret the command line 1/2

Hypothesis

```
Double.parseDouble(args[0])  
if ("+" == args[1])
```

Experiment

Calculator.java

```
public class Calculator {  
  
    public static void main(String[] args) {  
        double op1 = Double.parseDouble(args[0]);  
        double op2 = Double.parseDouble(args[2]);  
        if ("+" == args[1]) {  
            System.out.println(op1 + op2);  
        }  
    }  
}
```

No results

```
$ java Calculator 3.14 + 15
```



3. Interpret the command line 2/2

Hypothesis

*To compare objects we must use the
equals() method*

Experiment

Calculator.java

```
public class Calculator {  
  
    public static void main(String[] args) {  
        System.out.println(args[1]);  
        System.out.println("+ == args[1]);  
        System.out.println("+ .equals(args[1]));  
    }  
}
```

Hypothesis confirmed

```
$ java Calculator 3.14 + 15  
+  
false  
true
```



Calculator

Calculator.java

```
public class Calculator {  
  
    public static void main(String[] args) {  
        double op1 = Double.parseDouble(args[0]);  
        double op2 = Double.parseDouble(args[2]);  
        if ("+".equals(args[1])) {  
            System.out.println(op1 + op2);  
        } else if ("*".equals(args[1])) {  
            System.out.println(op1 * op2);  
        } else if ("/".equals(args[1])) {  
            System.out.println(op1 / op2);  
        } else if ("-".equals(args[1])) {  
            System.out.println(op1 - op2);  
        }  
    }  
}
```



Calculator using switch

Calculator.java

```
public class Calculator {
    public static void main(String[] args) {
        double op1 = Double.parseDouble(args[0]);
        double op2 = Double.parseDouble(args[2]);
        switch (args[1]) {
            case "+":
                System.out.println(op1 + op2);
                break;
            case "*":
                System.out.println(op1 * op2);
                break;
            case "/":
                System.out.println(op1 / op2);
                break;
            case "-":
                System.out.println(op1 - op2);
                break;
        }
    }
}
```



Extended Calculator

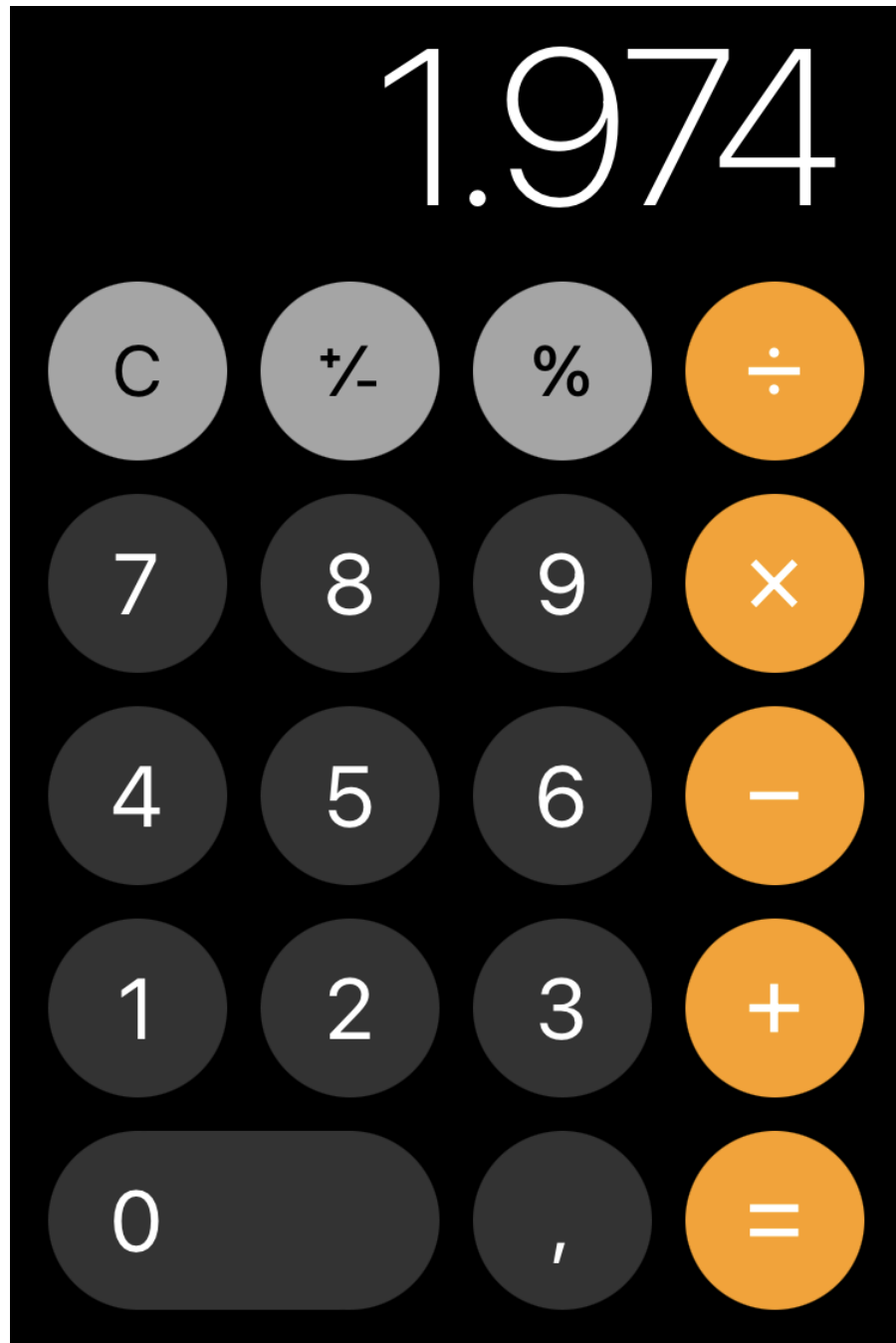
Calculator.java

```
public class Calculator {
    public static void main(String[] args) {
        var value = Double.parseDouble(args[0]);
        String operator = null;
        for (int i = 1; i < args.length; i++) {
            if (i % 2 == 1) {
                operator = args[i];
            } else {
                var operand = Double.parseDouble(args[i]);
                switch (operator) {
                    case "+":
                        value = value + operand;
                        break;
                    case "*":
                        value = value * operand;
                        break;
                    case "/":
                        value = value / operand;
                        break;
                    case "-":
                        value = value - operand;
                        break;
                }
            }
            System.out.println(value);
        }
    }
}
```

```
$ java Calculator.java 3.14 + 3.15 - 2 + 0.11
6.29
4.29
4.4
```



Assignment



Define a calculator class that

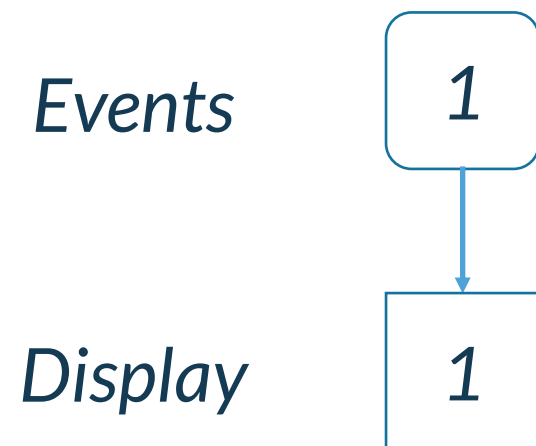
1. receives “events” from a calculator keyboard
2. sends the output to a *Display* object

```
class Display {  
    void display(String text) {  
        System.out.println(text);  
    }  
}
```

```
class Calculator {  
    final Display display;  
    //...  
    Calculator(Display display) {  
        this.display = display;  
    }  
    void plusPressed() {  
        //...  
    }  
    void zeroPressed() {  
        //...  
    }  
    //...  
}
```



Start simple



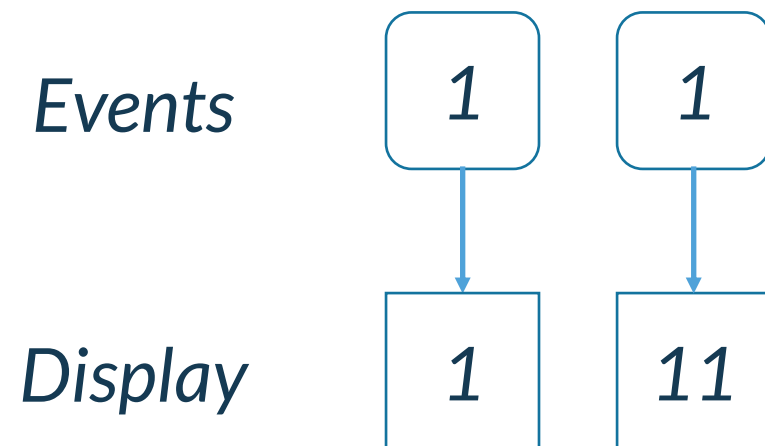
```
$ java Calculator.java  
1
```

Calculator.java

```
class CalculatorMain {  
    public static void main(String[] args) {  
        var calculator = new Calculator(new Display());  
        calculator.onePressed();  
    }  
}  
  
public class Calculator {  
  
    final Display display;  
  
    public Calculator(Display display) {  
        this.display = display;  
    }  
  
    void onePressed() {  
        display.display("1");  
    }  
}
```



Add one more event



```
$ java Calculator.java  
1  
11
```

Calculator.java

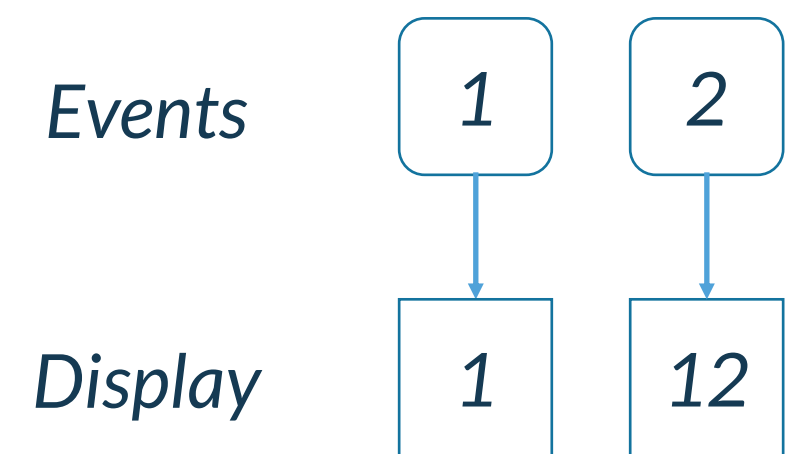
```
class CalculatorMain {  
    public static void main(String[] args) {  
        var calculator = new Calculator(new Display());  
        calculator.onePressed();  
        calculator.onePressed();  
    }  
}  
  
public class Calculator {  
  
    final Display display;  
    String string;  
  
    public Calculator(Display display) {  
        this.display = display;  
        string = "";  
    }  
  
    void onePressed() {  
        string += "1";  
        display.display(string);  
    }  
}
```



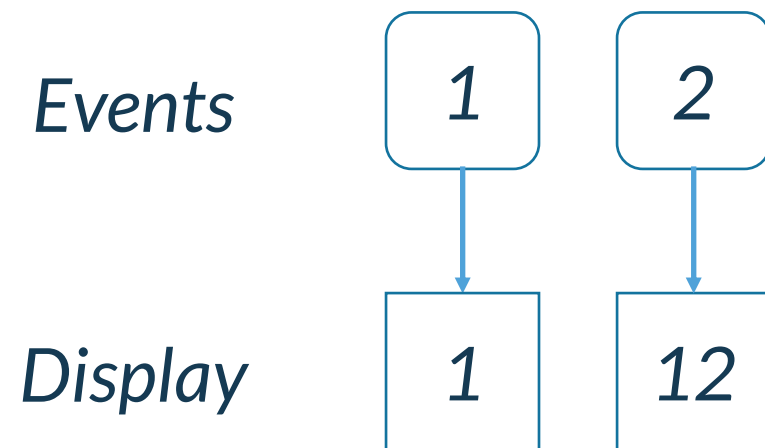
It seems to work



Let's extend the capabilities of our Calculator with an event of different type



Add one different event



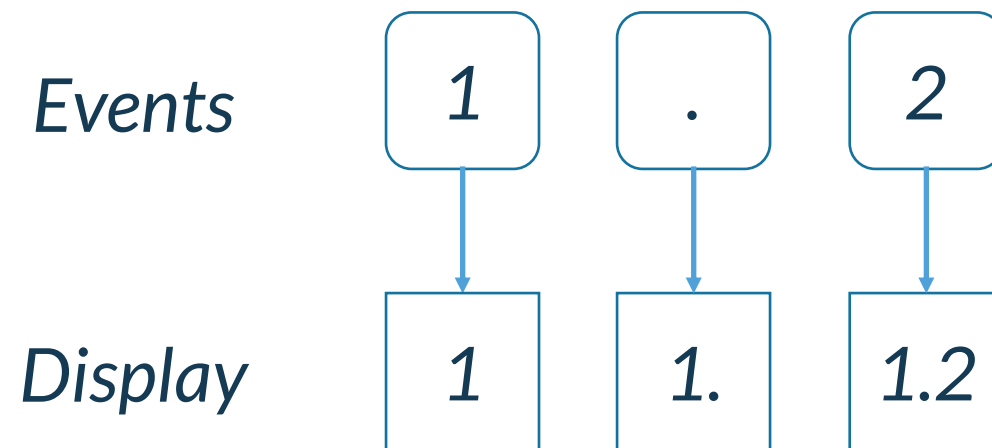
```
$ java Calculator.java  
1  
12
```

Calculator.java

```
class CalculatorMain {  
    public static void main(String[] args) {  
        var calculator = new Calculator(new Display());  
        calculator.onePressed();  
        calculator.twoPressed();  
    }  
}  
  
public class Calculator {  
    final Display display;  
    int string;  
  
    public Calculator(Display display) {  
        this.display = display;  
        string = "";  
    }  
  
    void onePressed() {  
        string += "1";  
        display.display(string);  
    }  
  
    void twoPressed() {  
        string += "2";  
        display.display(string);  
    }  
}
```



Dealing with dots



```
$ java Calculator.java
1
1.
1.2
```

Calculator.java

```
class CalculatorMain {
    public static void main(String[] args) {
        var calculator = new Calculator(new Display());
        calculator.onePressed();
        calculator.dotPressed();
        calculator.twoPressed();
    }
}

public class Calculator {
    final Display display;
    int string;

    public Calculator(Display display) {
        this.display = display;
        string = "";
    }

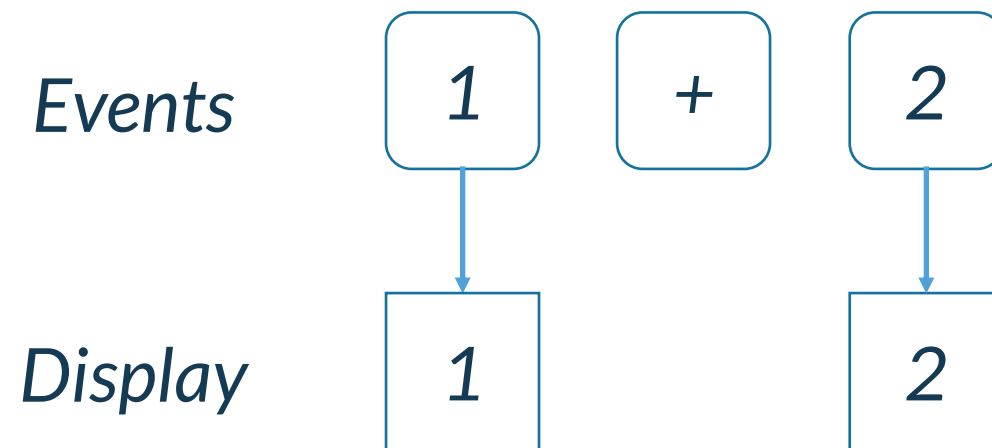
    void onePressed() {
        string += "1";
        display.display(string);
    }

    void twoPressed() {
        string += "2";
        display.display(string);
    }

    void dotPressed() {
        string += ".";
        display.display(string);
    }
}
```



Introduce operators



Calculator.java

```
class CalculatorMain {
    public static void main(String[] args) {
        var calculator = new Calculator(new Display());
        calculator.onePressed();
        calculator.plusPressed();
        calculator.twoPressed();
    }
}
```

```
$ java Calculator.java
1
2
```

Calculator.java

```
public class Calculator {
    final Display display;
    int string;

    public Calculator(Display display) {
        this.display = display;
        string = "";
    }

    void onePressed() {
        string += "1";
        display.display(string);
    }

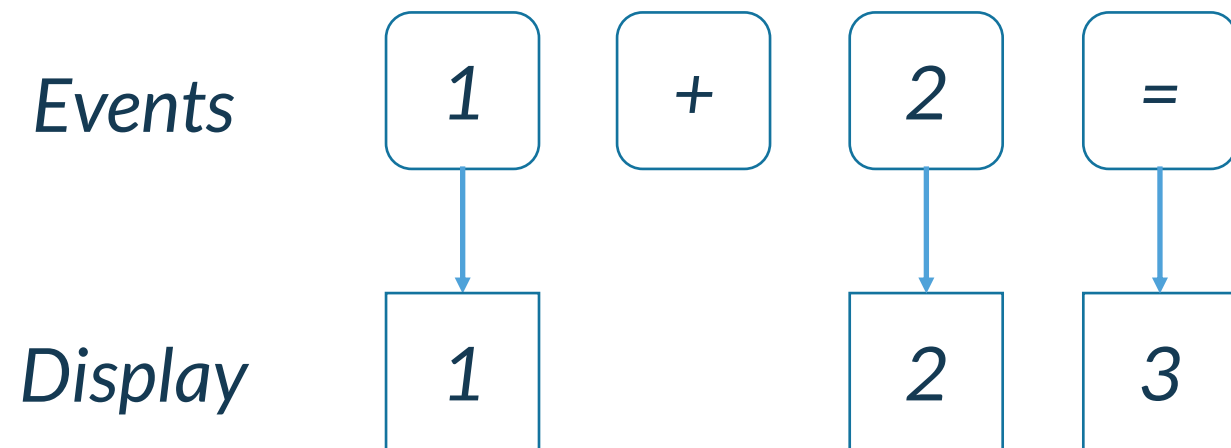
    void twoPressed() {
        string += "2";
        display.display(string);
    }

    void dotPressed() {
        string += ".";
        display.display(string);
    }

    void plusPressed() {
        string = "";
    }
}
```



Introduce equal



Calculator.java

```
class CalculatorMain {
    public static void main(String[] args) {
        var calculator = new Calculator(new Display());
        calculator.onePressed();
        calculator.plusPressed();
        calculator.twoPressed();
        calculator.equalPressed();
    }
}
```

```
$ java Calculator.java
1
2
3.0
```

Calculator.java

```
public class Calculator {
    final Display display;
    String string, operator;
    double op1;

    public Calculator(Display display) {
        this.display = display; string = "";
    }

    void onePressed() {
        string += "1"; display.display(string);
    }

    void twoPressed() {
        string += "2"; display.display(string);
    }

    void dotPressed() {
        string += "."; display.display(string);
    }

    void plusPressed() {
        op1 = Double.parseDouble(string);
        operator = "+"; string = "";
    }

    void equalPressed() {
        if ("+" .equals(operator)) {
            double op2 = Double.parseDouble(string);
            double result = op1 + op2;
            display.display("" + result); operator = "";
        }
    }
}
```



What is missing?

A lot of scenarios

When we turn on the Calculator it must show "0"

We can add multiple dots



Operations are not concatenated



Typing numbers after result evaluation



and many more

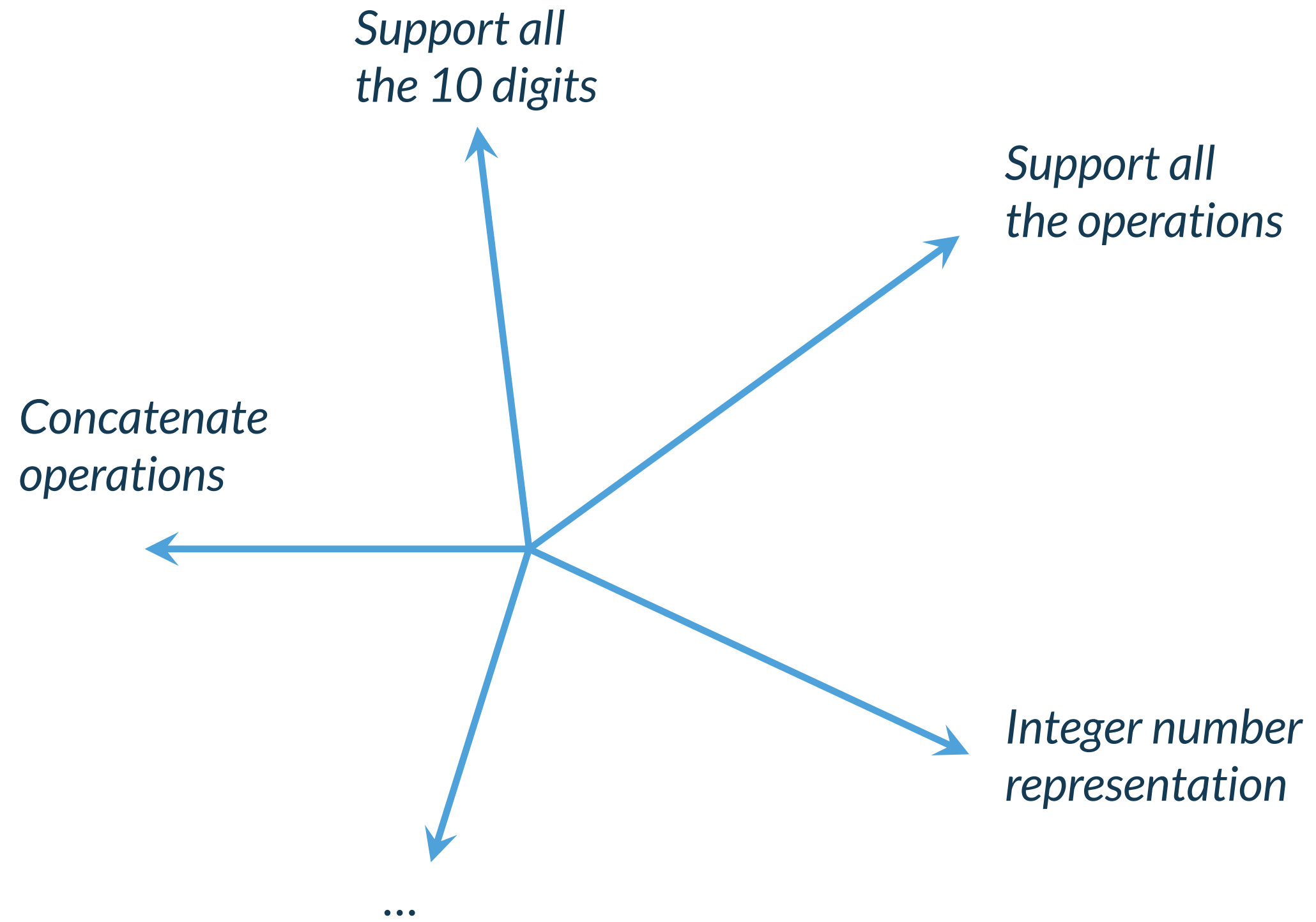


What have we learnt

- ✓ *Even an apparently trivial problem can become quickly very complex*
- ✓ *Different usage scenarios may trigger different behaviors*
- ✓ *The design of the solution is not unique and it not obvious from the beginning*
- ✓ *A problem should be decomposed over different directions*
- ✓ *An incremental approach can help developing the solution*



Dealing with complexity



Incremental development

1. *Select a usage scenario*
2. *Implement it*
3. *Don't forget to test all the previous scenarios*

More on point 3 in the second part of the course.





Thank you!

esteco.com

