JCP EC Education WG

Ken Fogel

April 13, 2020

Updated for April 23, 2020

Java Language Enhancement

Reviewed all changes since Java 8

Many enhancements addressed performance and security

Not enough addressed ease of teaching

I hope this is changing

Let’s look at what I have identified as important to education

JShell - Read-Evaluate-Print Loop (REPL) JDK 9

Hearlded as a tool for simplifying instruction

Execution as you enter code and press return

Immediate response line by line can be overwhelming/confusing

Writing entire methods first and immediately executing them more useful

Demo

JEP 330 - Launch Single-File Source-Code Programs JDK 11

- Addresses the overhead of running code
  - Traditional Style
    - Two-step to execution
      - javac
      - java -jar
  - Single-File Source-Code Style
    - One-step to execution
      - java
        - If the file has a public class with a main it compiles and executes
        - Works with preview features as well as established features
  - Demo

var – reduction of redundancy reduction

No more:	• `MyClass m = new MyClass();`
It now becomes:	• `var m = new MyClass();`
Encourages only creating objects with initialization	• Should reduce NPE

text blocks

Finally, what you enter into your source code is what you get

Especially useful for Strings that contain HTML, XML and JSON

Who doesn’t like writing three quotation marks in a row

"""	"""

switch – an expression & without a break

The best switch ever was in Microsoft Visual Basic 6

Not there yet but this is significant

A switch can return a value so no need to repeat ‘variable =‘ in every case

The end of break

- Remnant of C and machine language
- All cases terminate!

records – boilerplate reduction with immutable flavouring and a dash of compact constructor

Who likes writing for data objects:

- Initializing constructors, setters, getters, equals, hashCode, and toString
- NO ONE!

To the rescue is the immutable record

More than just a simplification of a bean

It’s the path to objects defaulting to immutability

And then there is the compact constructor

- Validating initial values without a separate constructor

What’s Pushing Java Aside?

JavaScript

- Nothing to download
- Available in the browsers on every school PC
- Numerous online programming environments

Python

- Associated with the two big trends:
  - Big Data
  - AI/ML
- Online Jupyter notepad is popular

Why is Python Gaining Popularity In Education?

My guess is its embrace of the vilest design pattern:

- Stream of Consciousness

I know I’m exaggerating

What I do see is its appeal to teachers who need to teach coding but don’t necessarily want to learn to code

Let’s Compare Python to Java

On the next slides are three variants of the same program in Python and Java

These programs request three floating point values

- Amount of money borrowed called the loan
- The annual percentage rate (APR) for interest on the borrowed money
- The length of the load expressed in months called the term

From these values the program calculates the monthly repayment and displays it

```
loan = input("            loan: ")
interest = input("       interest: ")
term = input("            term: ")
tempInterest = float(interest) / 12;
result = float(loan) * (tempInterest / (1.0 - ((1.0 + tempInterest) ** -float(term))));
print("Monthly Payment: %.2f" % result)
```

```
import java.util.Scanner;

public class JavaCalculator01 {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("          Loan: ");
        double loan = sc.nextDouble();
        System.out.print("      Interest: ");
        double interest = sc.nextDouble();
        System.out.print("          Term: ");
        double term = sc.nextDouble();
        double tempInterest = interest / 12.0;
        double result = loan *
                    (tempInterest / (1.0 - Math.pow((1.0 + tempInterest), -term)));
        System.out.println("Monthly Payment: " + String.format("%.2f", result));
    }
}
```

```
def func_input():
    global loan
    loan = float(input("          loan: "))
    global interest
    interest = float(input("      interest: "))
    global term
    term = float(input("          term: "))


def func_process():
    tempInterest = float(interest) / 12.0;
    global result
    result = loan * (tempInterest / (1.0 - ((1.0 + tempInterest) ** -term)));


def func_output():
    print('Monthly Payment: %.2f' % result)

func_input()
func_process()
func_output()
```

```
import java.util.Scanner;

public class JavaCalculator02 {

    static double loan;
    static double interest;
    static double term;
    static double result;

    private static void inputData() {
        Scanner sc = new Scanner(System.in);
        System.out.print("          Loan: ");
        loan = sc.nextDouble();
        System.out.print("      Interest: ");
        interest = sc.nextDouble();
        System.out.print("          Term: ");
        term = sc.nextDouble();
    }

    private static void processData() {
        double tempInterest = interest / 12.0;
        result = loan * (tempInterest / (1.0 - Math.pow((1.0 + tempInterest), -term)));
    }

    private static void outputResult() {
        System.out.println("Monthly Payment: " + String.format("%.2f", result));
    }

    public static void main(String[] args) {
        inputData();
        processData();
        outputResult();
    }
}
```

```
class PythonCalculator03:

    def func_input(self):
        loan = float(input("          loan: "))
        interest = float(input("      interest: "))
        term = float(input("          term: "))
        return loan, interest, term

    def func_process(self, input_data):
        (loan, interest, term) = input_data
        temp_interest = float(interest) / 12.0;
        return loan * (temp_interest / (1.0 - ((1.0 + temp_interest) ** -term)));

    def func_output(self, result):
        print('Monthly Payment: %.2f' % result)

    def func_work(self):
        input_data = self.func_input()
        result = self.func_process(input_data)
        self.func_output(result)


worker = PythonCalculator03()
worker.func_work()
```

```java
import java.util.Scanner;

public class JavaCalculator03 {

    private LoanBean inputData() {
        Scanner sc = new Scanner(System.in);
        System.out.print("          Loan: ");
        double loan = sc.nextDouble();
        System.out.print("      Interest: ");
        double interest = sc.nextDouble();
        System.out.print("          Term: ");
        double term = sc.nextDouble();
        return new LoanBean(loan, interest, term);
    }

    private double processData(LoanBean loan) {
        double tempInterest = loan.interest() / 12.0;
        double result = loan.loan() * (tempInterest / (1.0 - Math.pow((1.0 + tempInterest), -loan.term())));
        return result;
    }

    private void outputResult(double result) {
        System.out.println("Monthly Payment: " + String.format("%.2f", result));
    }

    public void perform() {
        var loan = inputData();
        var result = processData(loan);
        outputResult(result);
    }

    public static void main(String[] args) {
        JavaCalculator03 calc = new JavaCalculator03();
        calc.perform();
    }
}

record LoanBean(double loan, double interest, double term) {
}
```

Some suggestions

- Create an input method that combines the prompt with the input
- Replace Math.pow with an operator
- Get rid of **public static void main(String[] args)**
- Promote development of Java in the AI/ML space such as JSR381
- Promote online learning sites for Java that are teacher friendly
- Promote resources for teachers
- Hold seminars for college and university instructors

Conclusion

- More than ever our society, our civilization needs programmers
  - Sorry for the hyperbole
- It has been suggested that computer programming should be a core course up there with reading, writing and arithmetic
- I'd rather have a programmer trained in Java take on Python than vice versa
- We need a strong outreach to schools, colleges and universities
- We need the corporate muscle of the JCP
- After all, students will be your employees sooner than you think