Complete Java Masterclass @Udemy.com

CareerDevs Classroom Presentation
December 18, 2017
@GeekyCoderr

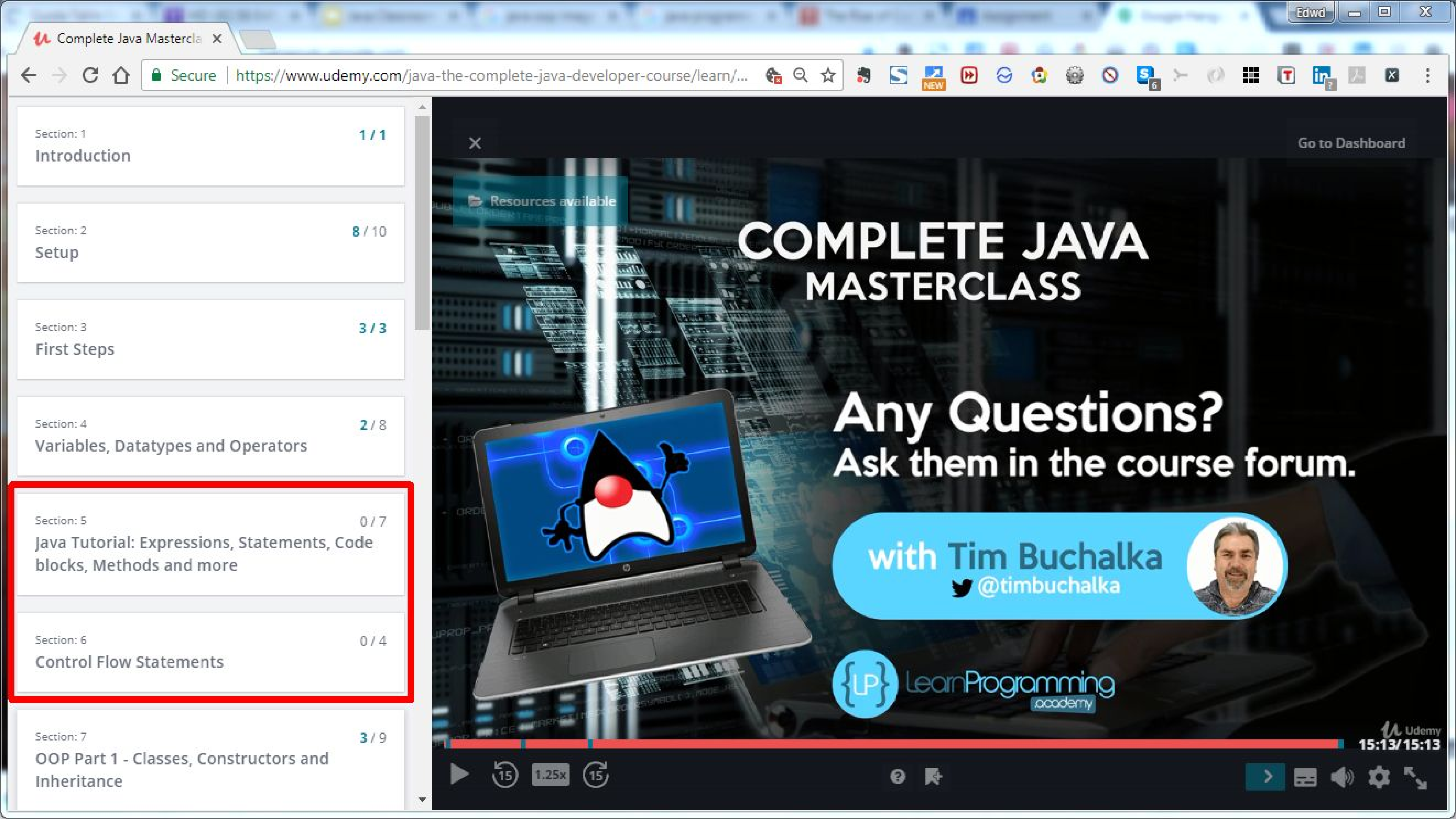
Brief Overview Section 5 & 6

The view from 10,000 feet up.

We started this tutorial a week ago.

How far did you get?

That’s a rhetorical question. You don’t have to tell me.



Section: 1
Introduction
1 / 1

Section: 2
Setup
8 / 10

Section: 3
First Steps
3 / 3

Section: 4
Variables, Datatypes and Operators
2 / 8

Section: 5
Java Tutorial: Expressions, Statements, Code blocks, Methods and more
0 / 7

Section: 6
Control Flow Statements
0 / 4

Section: 7
OOP Part 1 - Classes, Constructors and Inheritance
3 / 9

COMPLETE JAVA
MASTERCLASS

Any Questions?
Ask them in the course forum.

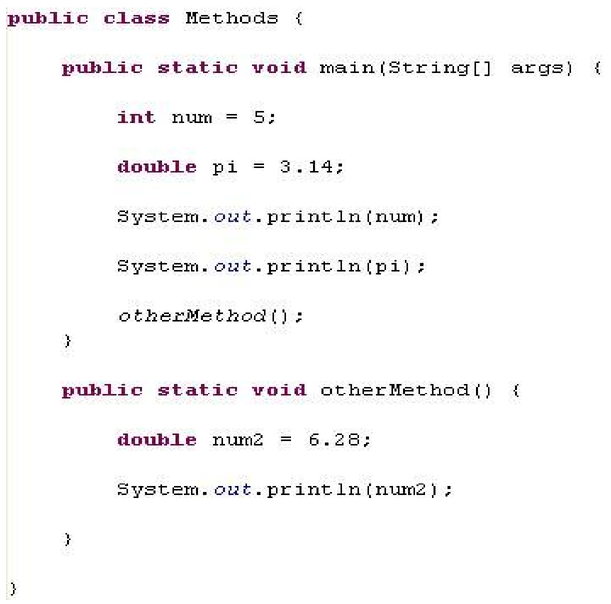
with Tim Buchalka
@timbuchalka

LearnProgramming
academy

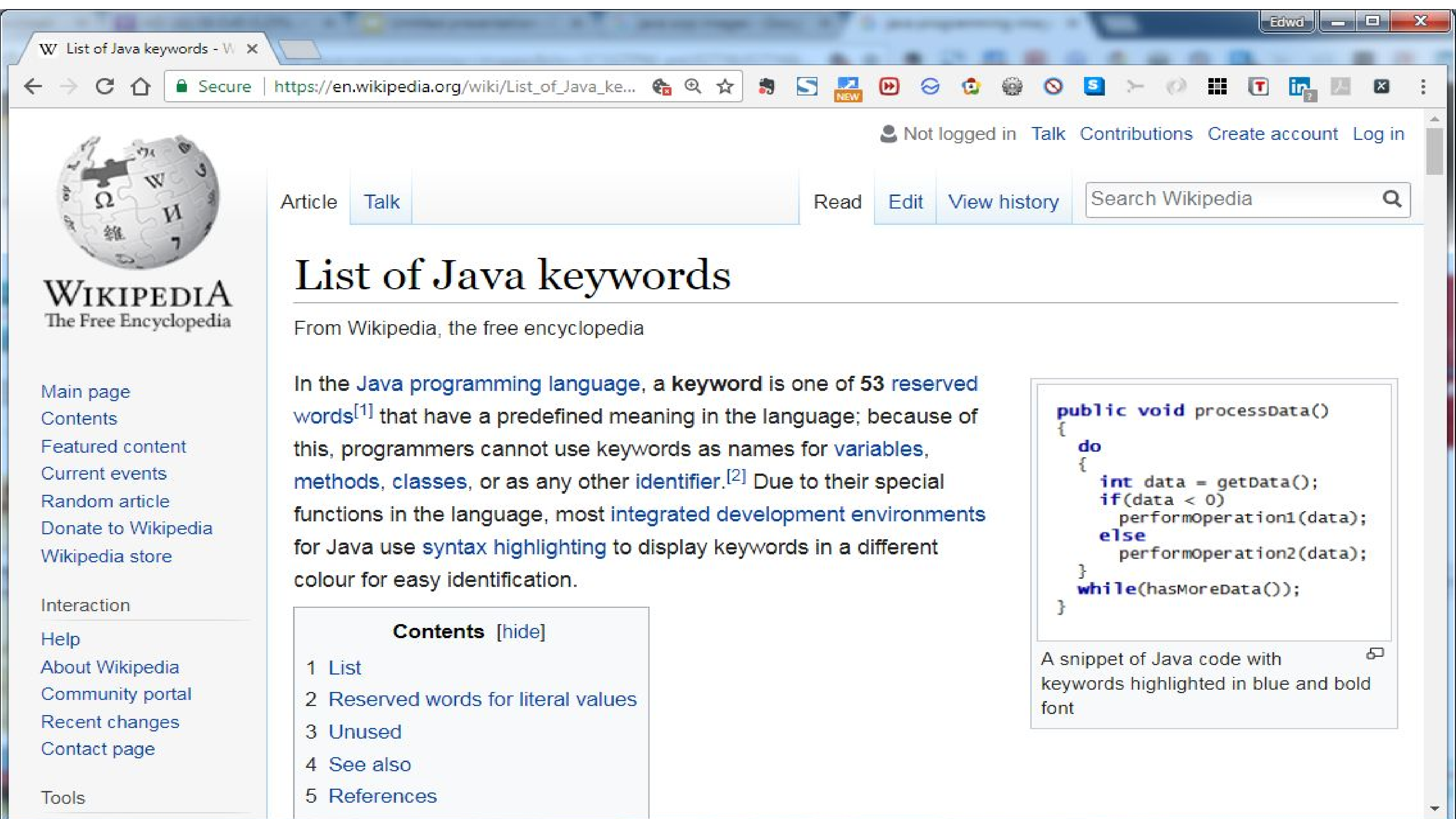
Before you can get to the fun stuff like OOP...

Just like a pastry chef has to learn how to measure and calculate before they make great yummy cakes...

We have to learn the boring stuff like Keywords, Expressions, Statements, Code blocks, Methods and more...



```java
public class Methods {

    public static void main(String[] args) {

        int num = 5;

        double pi = 3.14;

        System.out.println(num);

        System.out.println(pi);

        otherMethod();
    }

    public static void otherMethod() {

        double num2 = 6.28;

        System.out.println(num2);

    }

}
```



List of Java keywords

From Wikipedia, the free encyclopedia

In the Java programming language, a **keyword** is one of **53** reserved words[1] that have a predefined meaning in the language; because of this, programmers cannot use keywords as names for variables, methods, classes, or as any other identifier.[2] Due to their special functions in the language, most integrated development environments for Java use syntax highlighting to display keywords in a different colour for easy identification.

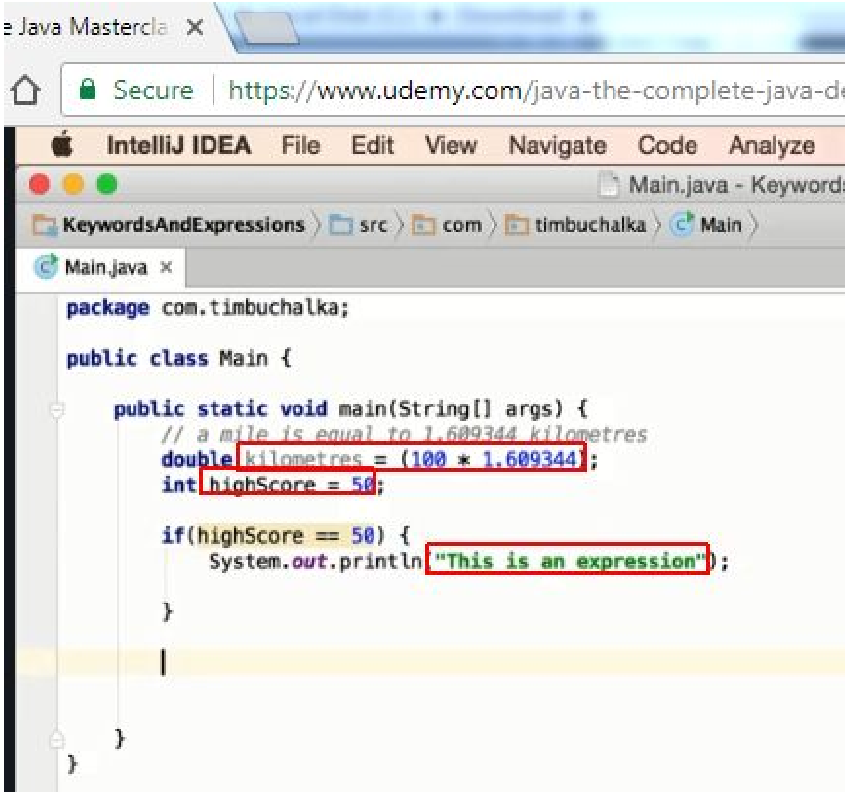
```
public void processData()
{
  do
  {
    int data = getData();
    if(data < 0)
      performOperation1(data);
    else
      performOperation2(data);
  }
  while(hasMoreData());
}
```

A snippet of Java code with keywords highlighted in blue and bold font

Contents [hide]

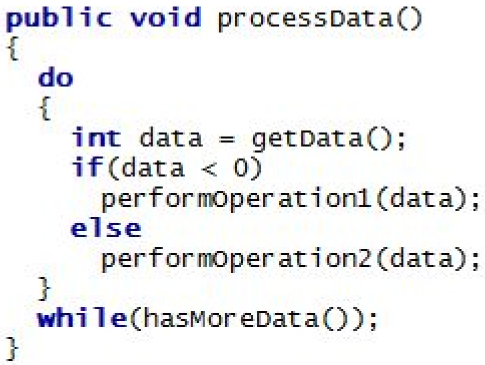
Expressions

- Between the datatype and semi-colon
  ex. double E = mc^2;
- Prefix and postfix operators
  ex. a++, b--, ++c, --d
- Inside the parentheses of a method
  println:System.out.println(‘expression”
- Object instantiation
  ex. Coffee myCoffee = new Coffee();
- Method calls with or w/out return value
  ex. boolean isSweet = addSugar(2);
- Ternary operators
  ex. myCoffee = Hazelnut ? iced : hot;

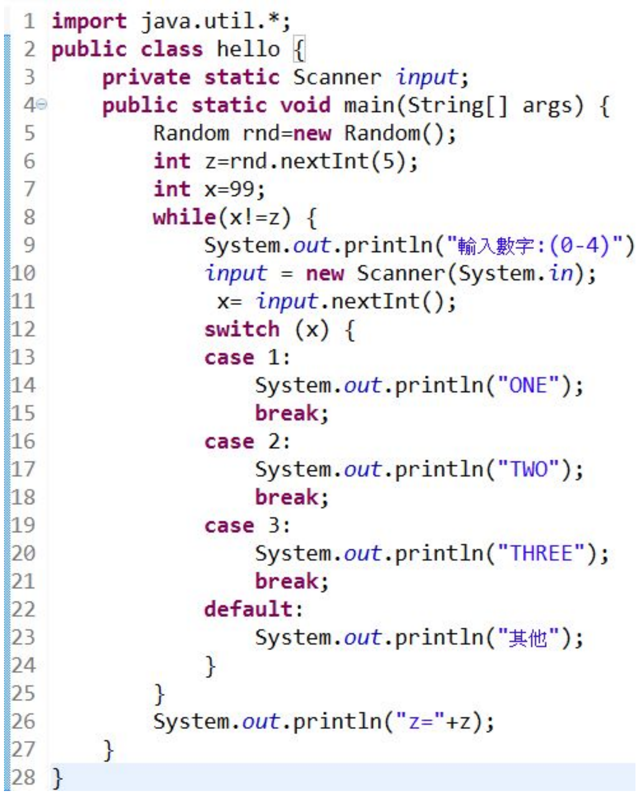


```
package com.timbuchalka;

public class Main {

    public static void main(String[] args) {
        // a mile is equal to 1.609344 kilometres
        double kilometres = (100 * 1.609344);
        int highScore = 50;

        if(highScore == 50) {
            System.out.println("This is an expression");
        }

    }
}
```

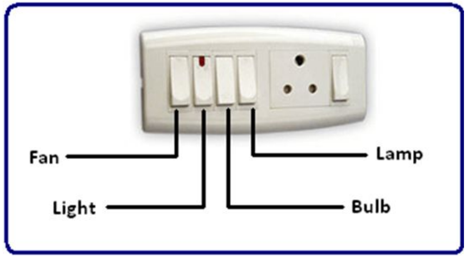
Control Flow Statements

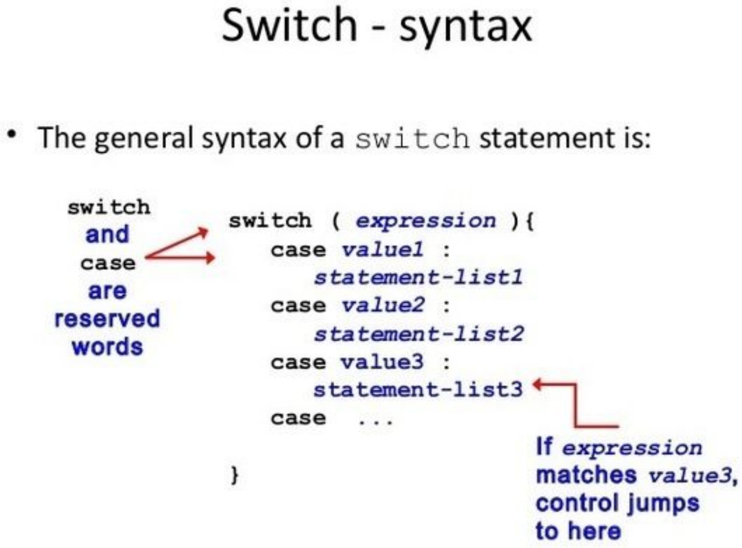


```
public void processData()
{
  do
  {
    int data = getData();
    if(data < 0)
      performOperation1(data);
    else
      performOperation2(data);
  }
  while(hasMoreData());
}
```



```
import java.util.*;
public class hello {
    private static Scanner input;
    public static void main(String[] args) {
        Random rnd=new Random();
        int z=rnd.nextInt(5);
        int x=99;
        while(x!=z) {
            System.out.println("輸入數字:(0-4)")
            input = new Scanner(System.in);
             x= input.nextInt();
            switch (x) {
            case 1:
                System.out.println("ONE");
                break;
            case 2:
                System.out.println("TWO");
                break;
            case 3:
                System.out.println("THREE");
                break;
            default:
                System.out.println("其他");
            }
        }
        System.out.println("z="+z);
    }
}
```





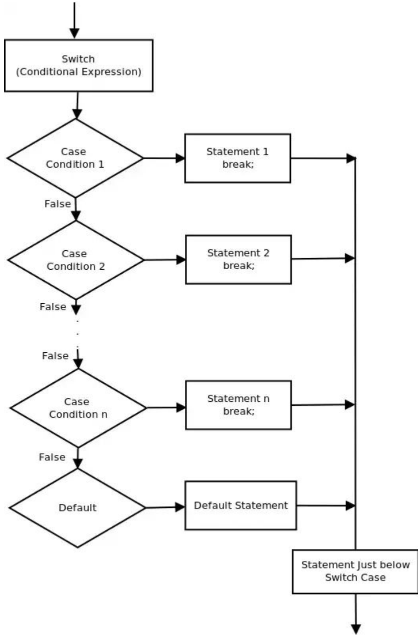
Switch - syntax

- The general syntax of a switch statement is:

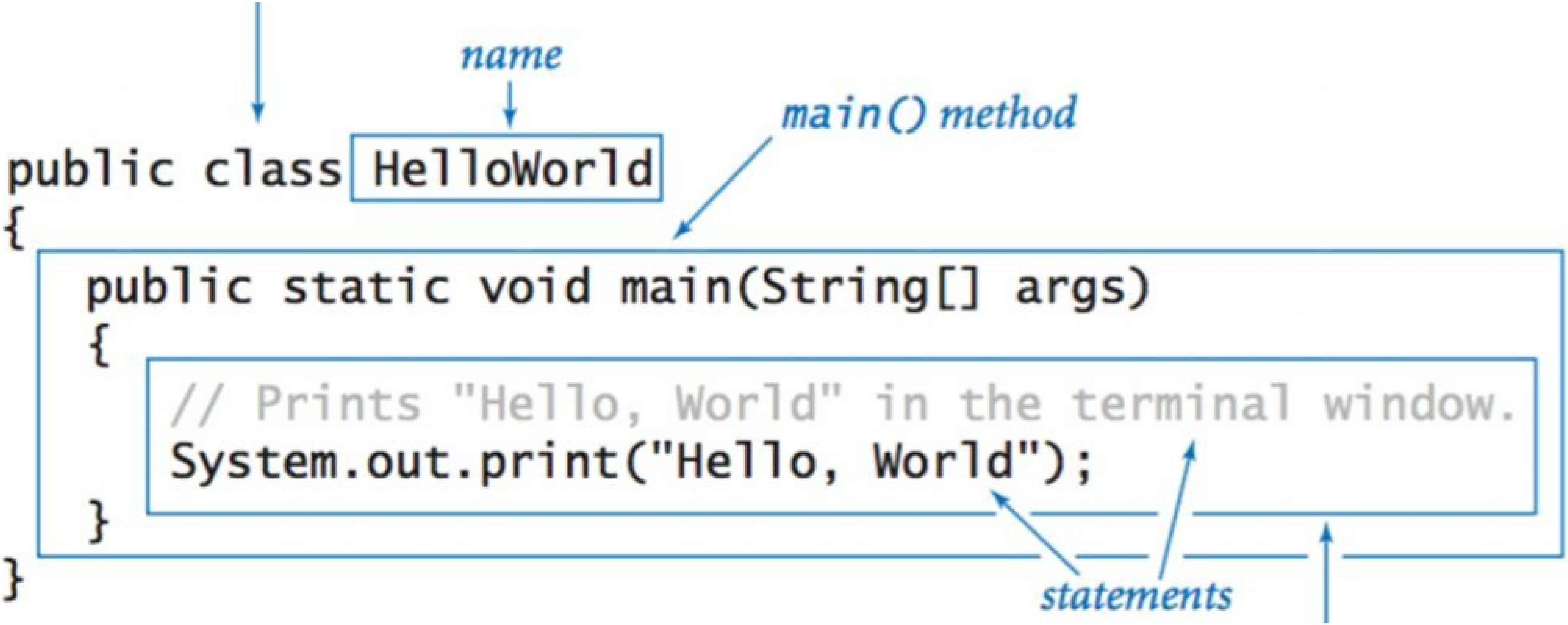
switch and case are reserved words

```
switch ( expression ){
   case value1 :
      statement-list1
   case value2 :
      statement-list2
   case value3 :
      statement-list3
   case ...

}
```

If *expression* matches *value3*, control jumps to here



Statements, Code Blocks, Methods and more...

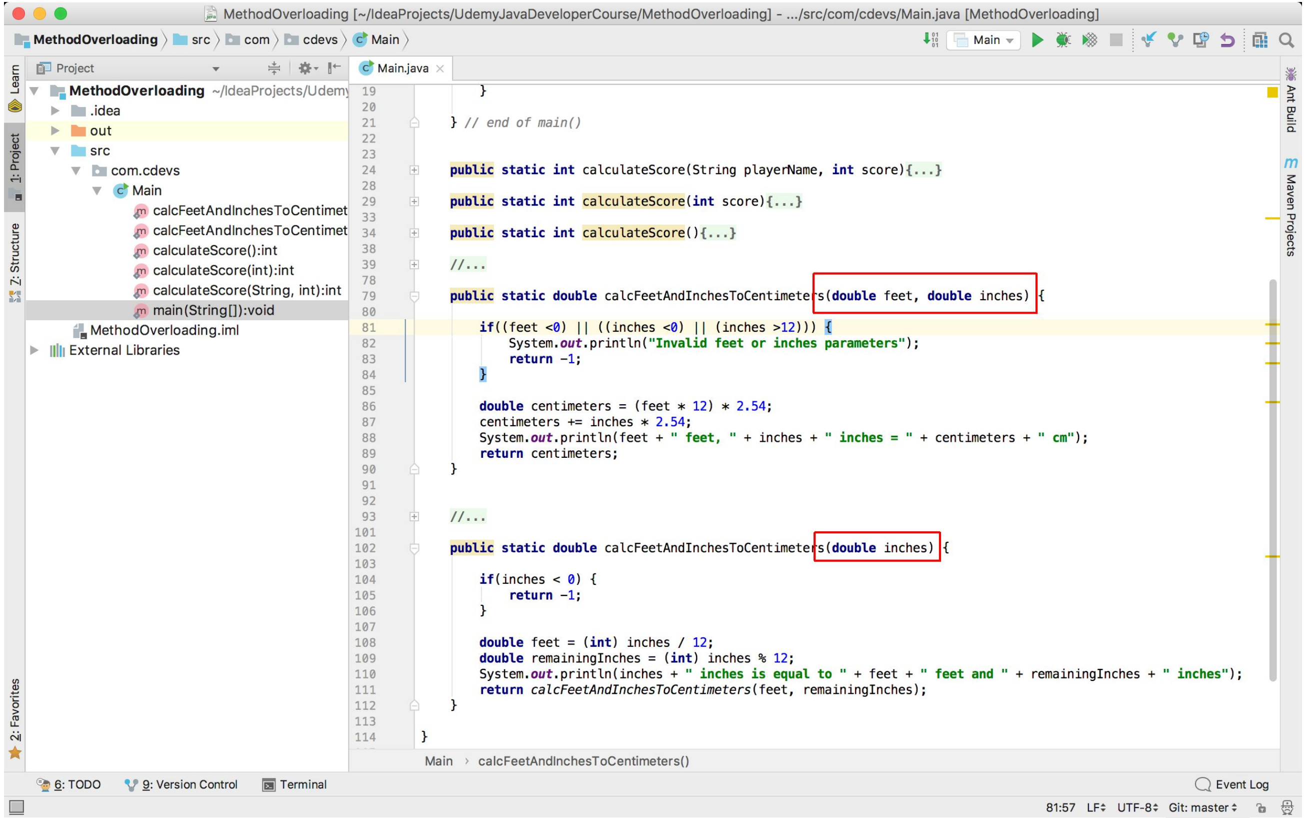


```
public class HelloWorld
{
   public static void main(String[] args)
   {
      // Prints "Hello, World" in the terminal window.
      System.out.print("Hello, World");
   }
}
```

Methods and Method Overloading

Overloaded Methods:

- MUST have different number and/or type of parameters.
- CAN have different return type
- CAN have different access modifier

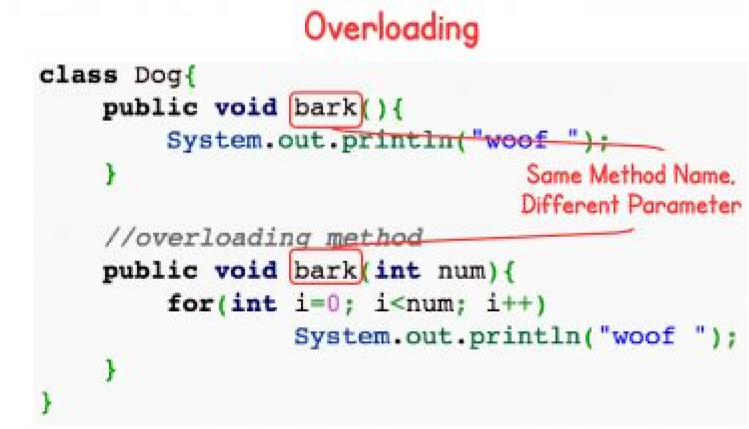


```java
        }

    } // end of main()


    public static int calculateScore(String playerName, int score){...}

    public static int calculateScore(int score){...}

    public static int calculateScore(){...}

    //...

    public static double calcFeetAndInchesToCentimeters(double feet, double inches) {

        if((feet <0) || ((inches <0) || (inches >12))) {
            System.out.println("Invalid feet or inches parameters");
            return -1;
        }

        double centimeters = (feet * 12) * 2.54;
        centimeters += inches * 2.54;
        System.out.println(feet + " feet, " + inches + " inches = " + centimeters + " cm");
        return centimeters;
    }


    //...

    public static double calcFeetAndInchesToCentimeters(double inches) {

        if(inches < 0) {
            return -1;
        }

        double feet = (int) inches / 12;
        double remainingInches = (int) inches % 12;
        System.out.println(inches + " inches is equal to " + feet + " feet and " + remainingInches + " inches");
        return calcFeetAndInchesToCentimeters(feet, remainingInches);
    }

}
```

Novice Java programmers often confuse

Method Overloading vs. Overriding

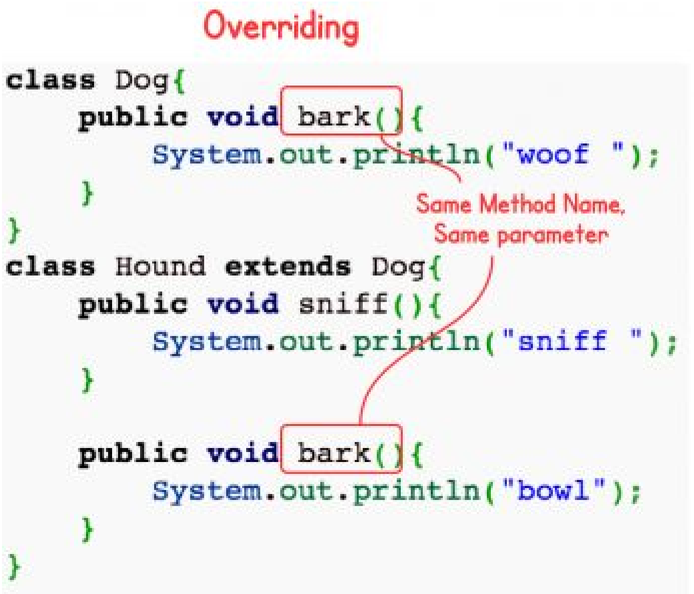
https://www.programcreek.com/2009/02/overriding-and-overloading-in-java-with-examples/



Overloading

```
class Dog{
    public void bark(){
        System.out.println("woof ");
    }

    //overloading method
    public void bark(int num){
        for(int i=0; i<num; i++)
            System.out.println("woof ");
    }
}
```

Same Method Name,
Different Parameter



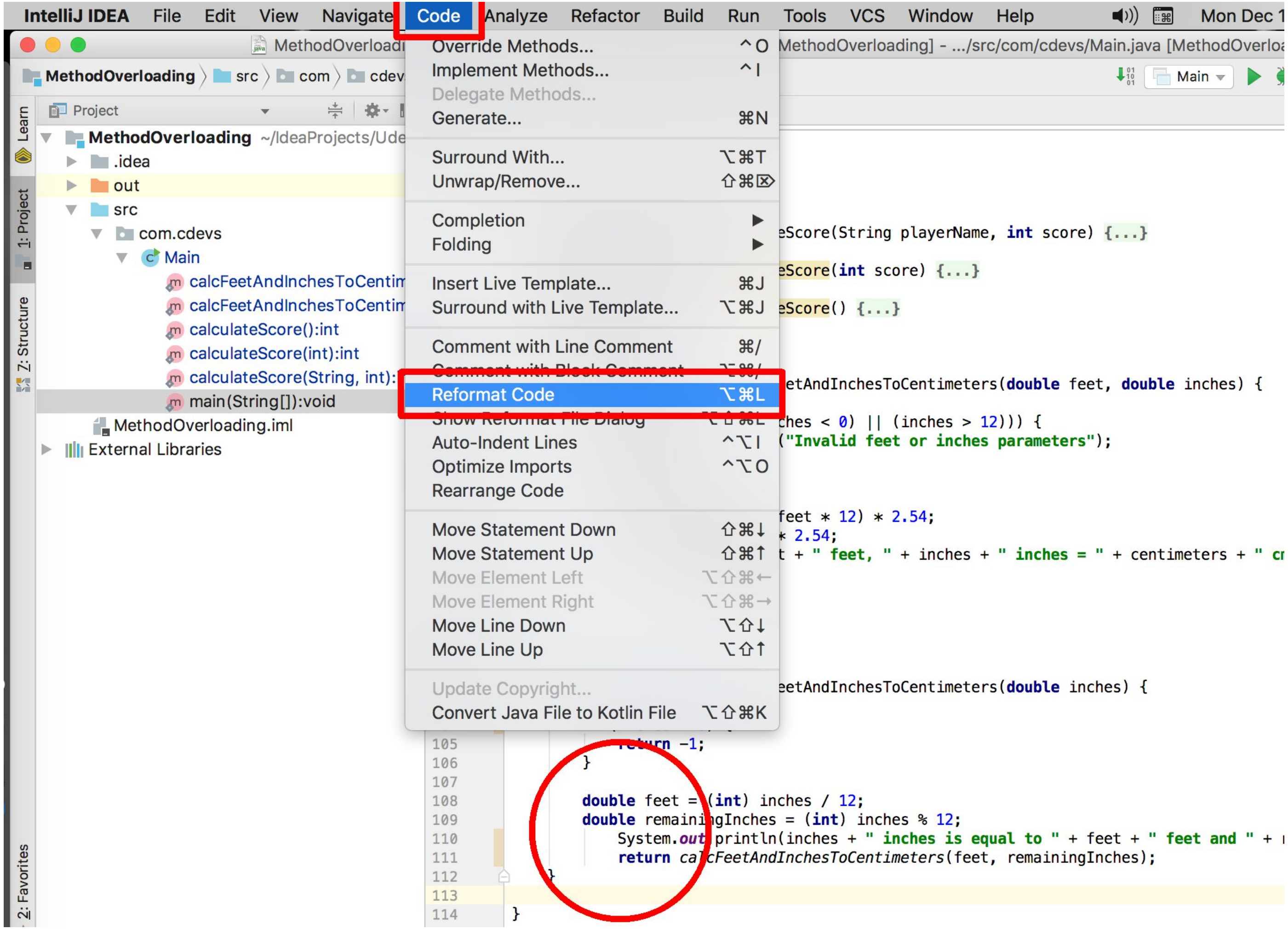
Overriding

```
class Dog{
    public void bark(){
        System.out.println("woof ");
    }
}
class Hound extends Dog{
    public void sniff(){
        System.out.println("sniff ");
    }

    public void bark(){
        System.out.println("bowl");
    }
}
```

Same Method Name,
Same parameter

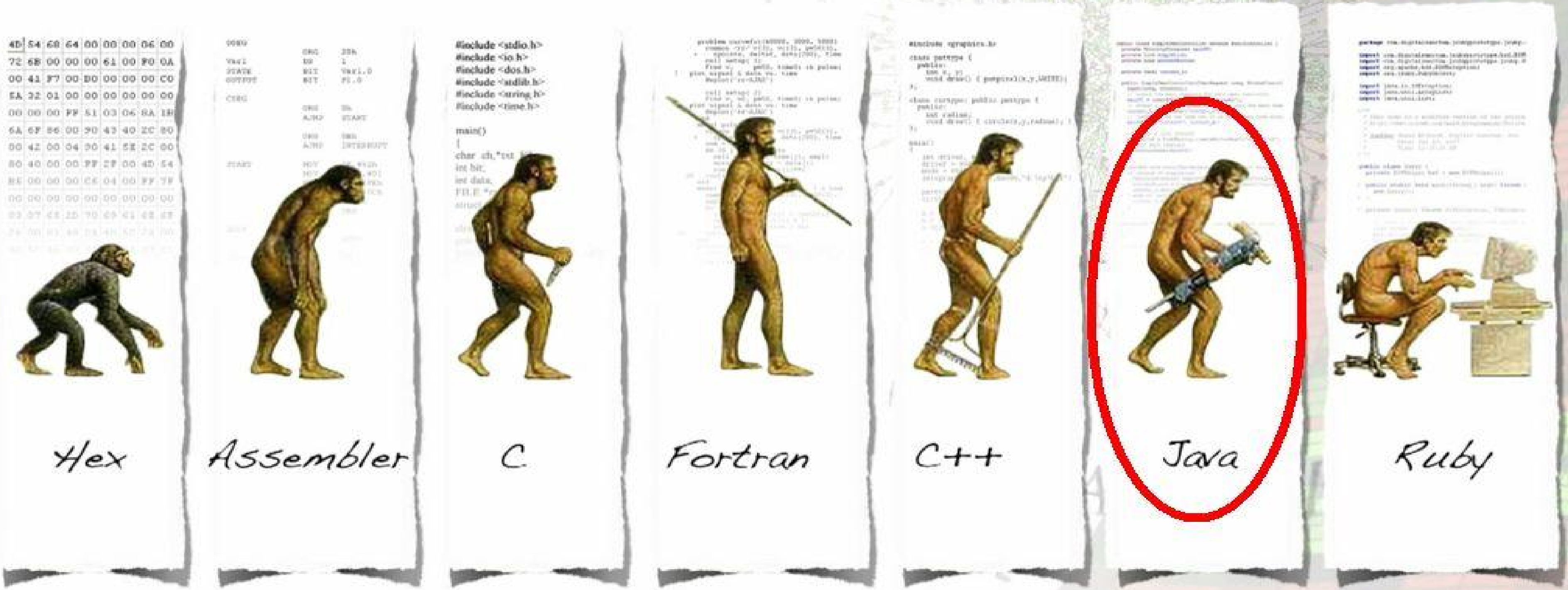
Reformat Code

for when your typing gets out of control and you want to tidy up a bit before you commit and push to GitHub



```
eScore(String playerName, int score) {...}

eScore(int score) {...}

eScore() {...}

etAndInchesToCentimeters(double feet, double inches) {
ches < 0) || (inches > 12))) {
("Invalid feet or inches parameters");

feet * 12) * 2.54;
* 2.54;
t + " feet, " + inches + " inches = " + centimeters + " cr

eetAndInchesToCentimeters(double inches) {

            return -1;
        }

        double feet = (int) inches / 12;
        double remainingInches = (int) inches % 12;
            System.out.println(inches + " inches is equal to " + feet + " feet and " + i
            return calcFeetAndInchesToCentimeters(feet, remainingInches);
    }

}
```

The Evolution Of Computer Programming Languages


Hex
Assembler
C
Fortran
C++
Java
Ruby