Java Syntax

Term 2 - Week 1 - Academic Year 2022-23

Overview

- Introduction to Java
- Syntax
- Workshop

Introduction to Java

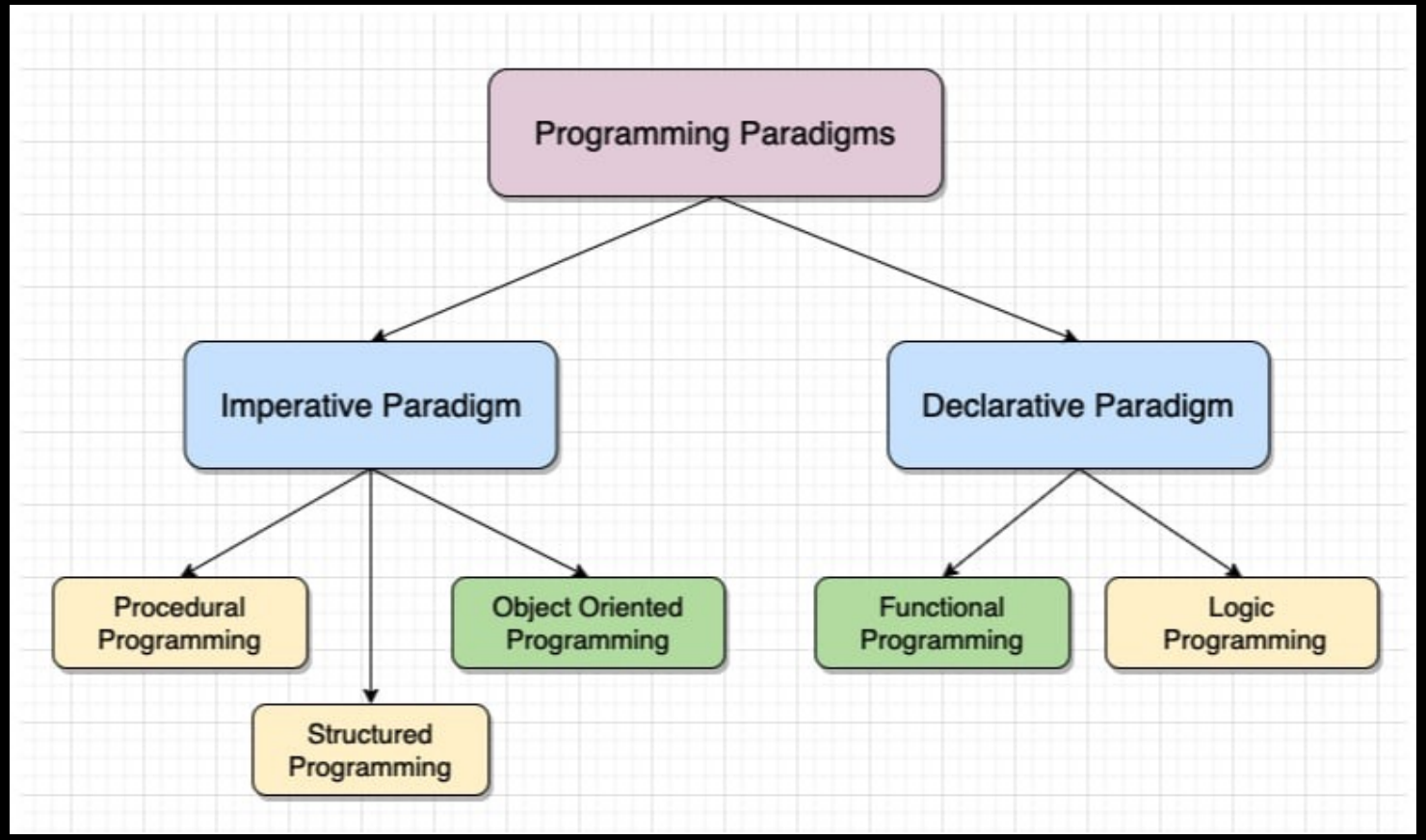
Java and the JVM

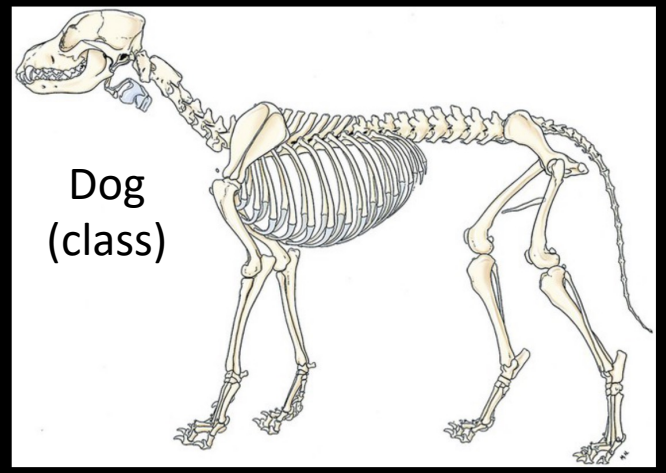
Java is a **purely object-oriented language (OOP)**. Every class is its own file, and no code can exist outside of a class.

Java binary is executed by the **Java Virtual Machine (JVM)**, so it can run on any OS.

Object-Oriented Programming (OOP)

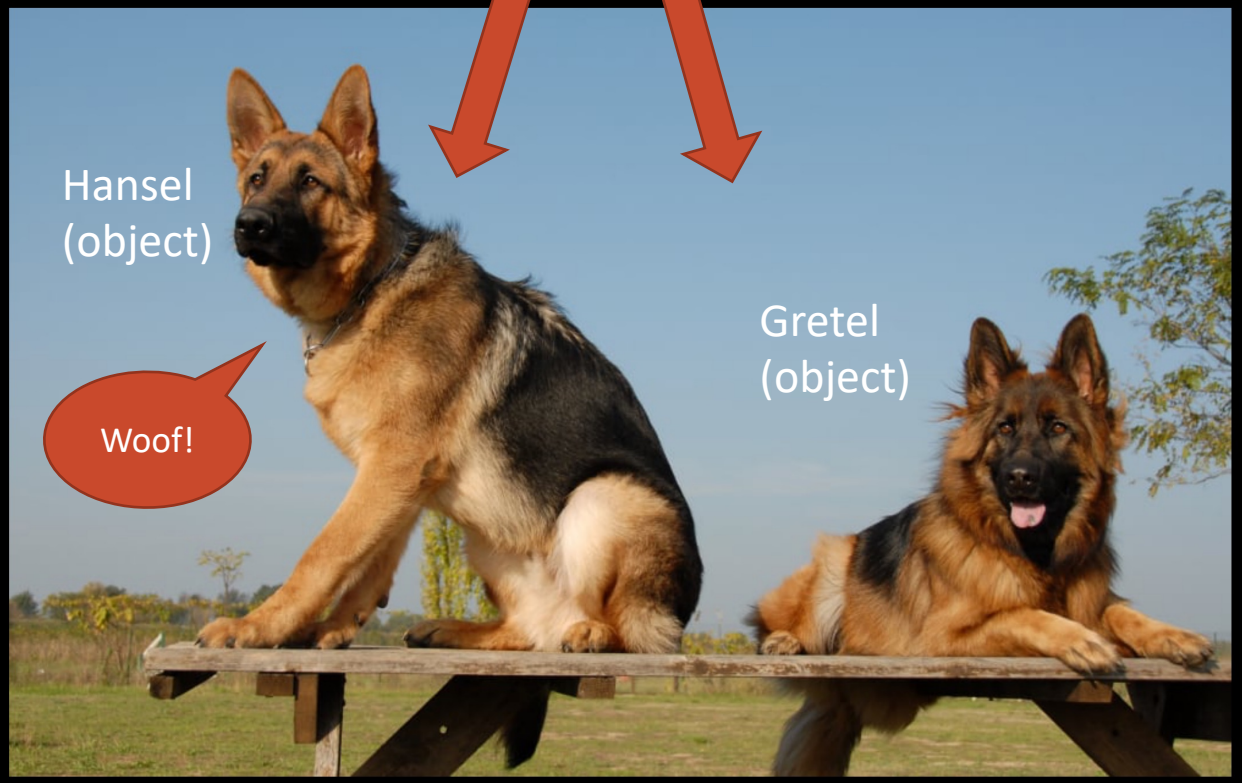
OOP languages are **fully based on objects** which have different **functionalities**.






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

    public void run(){
        System.out.println("The owner lost the dog.");
    }
}
```



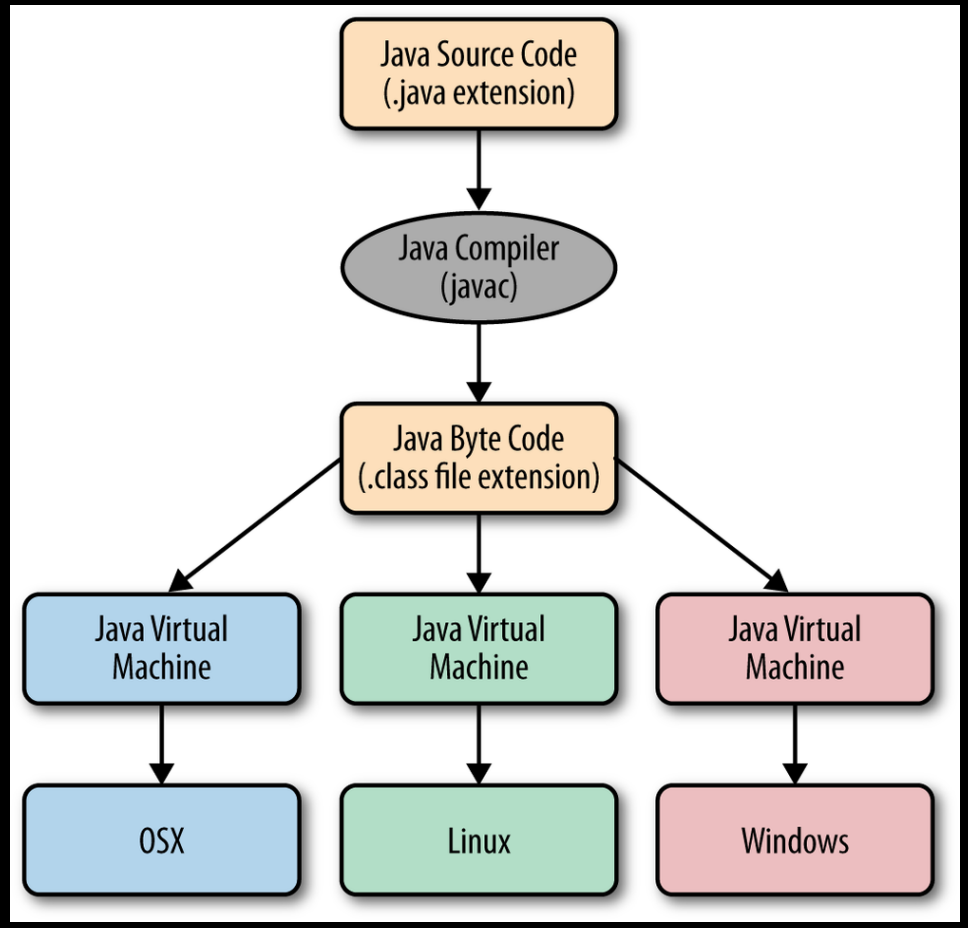

```java
public class Main {
    public static void main(String[] args){
        Dog Hansel = new Dog();
        Dog Gretel = new Dog();
        Hansel.bark();
    }
}
```

IntelliJ

Helpful Java-specific IDE which abstracts away more complex parts of the language.

	
• Consistent refactoring tips • Seamless integration of several tools (e.g., Ant, Maven, etc.)	• Slow to start • Hand-holds the developer

Compilation & Execution



Compilation & Execution

IntelliJ:

^ ⇨ R

Terminal:

Compile Java code → `javac -d <out_dir> <src_dir>;`

```
cd <out_dir>;
```

Execute Java binary → `java Main.class;`

Syntax

Fundamentals

No code can exist outside of a Java class, and every Java class must exist within a **single file**.

Executable Java classes need a **main() method.**

```
public class Main {
    public static void main(String[] args){
        System.out.println("Hello UCL!");
    }
}
```

Each general scope is delimited by **curved** brackets, and each conditional is delimited by **regular** brackets.

Variables, Objects, & References

Variables are named storage of primitive types.

```
int a = 1; // Primitive variable
Object b = new Object(); // Object instantiation
```

Reference

Object

Objects are instances of non-primitive data types or classes.

References are variables that points towards the object's memory address.

Conditionals, Control Flow, & Loops

If-else statements

```
if (this.name.equals(name)){
     System.out.println("Your cat has arrived.");
} else if (this.name.equals("Doraemon")){
     System.out.println("That's a robot, dummy!");
} else {
     System.out.println("Nothing happened.");
}
```

Switch cases

```
switch (title){
     case "Get Out":
          System.out.println("Blumhouse Productions");
          break;
     case "The Whale":
          System.out.println("A24");
          break;
     default:
          System.out.println("Marvel");
}
```

Conditionals, Control Flow, & Loops

For loop

```
for (int i = 0; i<5; i++){
    System.out.println(i);
}
```

While loop

```
int i = 0;
while (i < 5){
    System.out.println(i);
    i++;
}
```

Do-while loop

```
int i = 0;
do {
    System.out.println(i);
    i++;
} while (i < 5);
```

Classes

A class consists of a **constructor**, **methods** (i.e. functions), and **variables**.

Methods and variables can be **static** (class-owned) or dependent on **instances** (object-owned).

```java
public class Cat {
    static String species = "Cat"; // Static class variable
    private final String name; // Private instance variable

    public Cat(String name){ // Custom Constructor
        this.name = name;
    }

    public static void meow(){ // Public static method
        System.out.println("Meow!");
    }

    public void call(String name){ // Public instance method
        if (this.name.equals(name)){
            System.out.println("Your cat has arrived.");
        } else {
            System.out.println("Nothing happened.");
        }
    }
}
```

Workshop

Given a string as input, find the longest substring with no repeating characters.

If you haven't done so already, fork this repository to add your solution:

https://github.com/Catcatcher33/programming-tutor-22-23