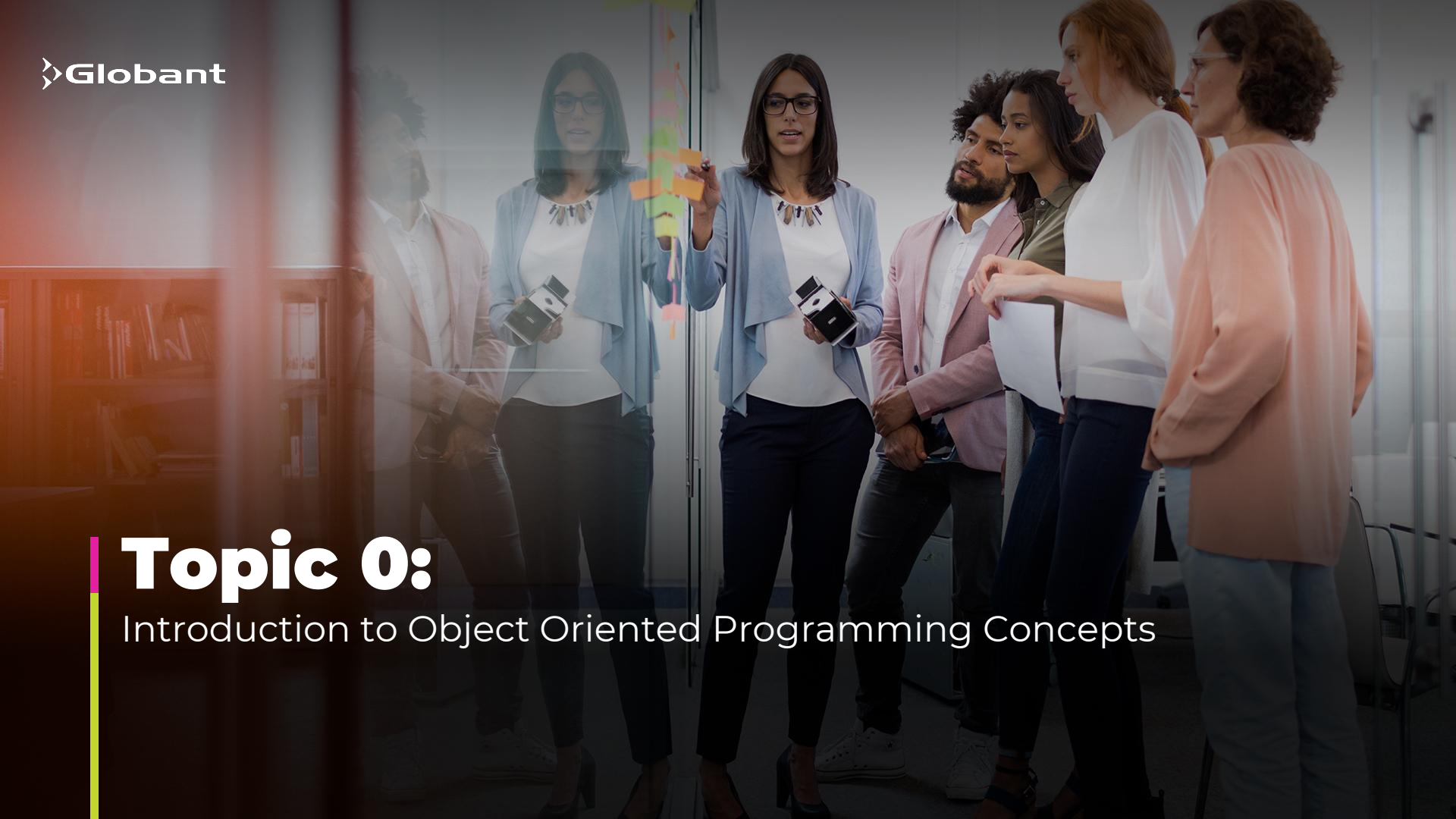


Topic 0:

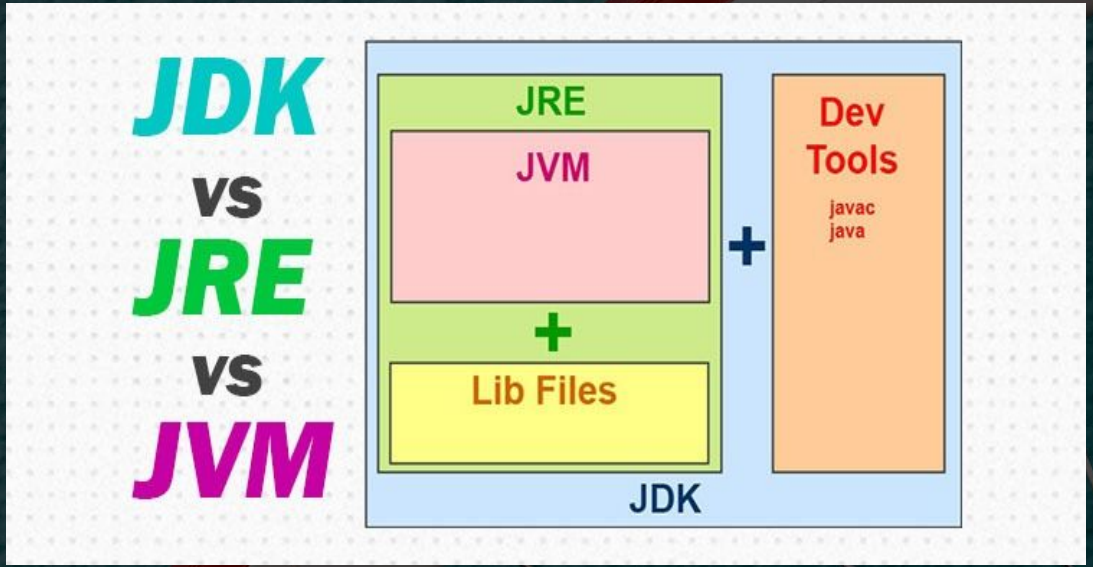
Introduction to Object Oriented Programming Concepts

What's Java?

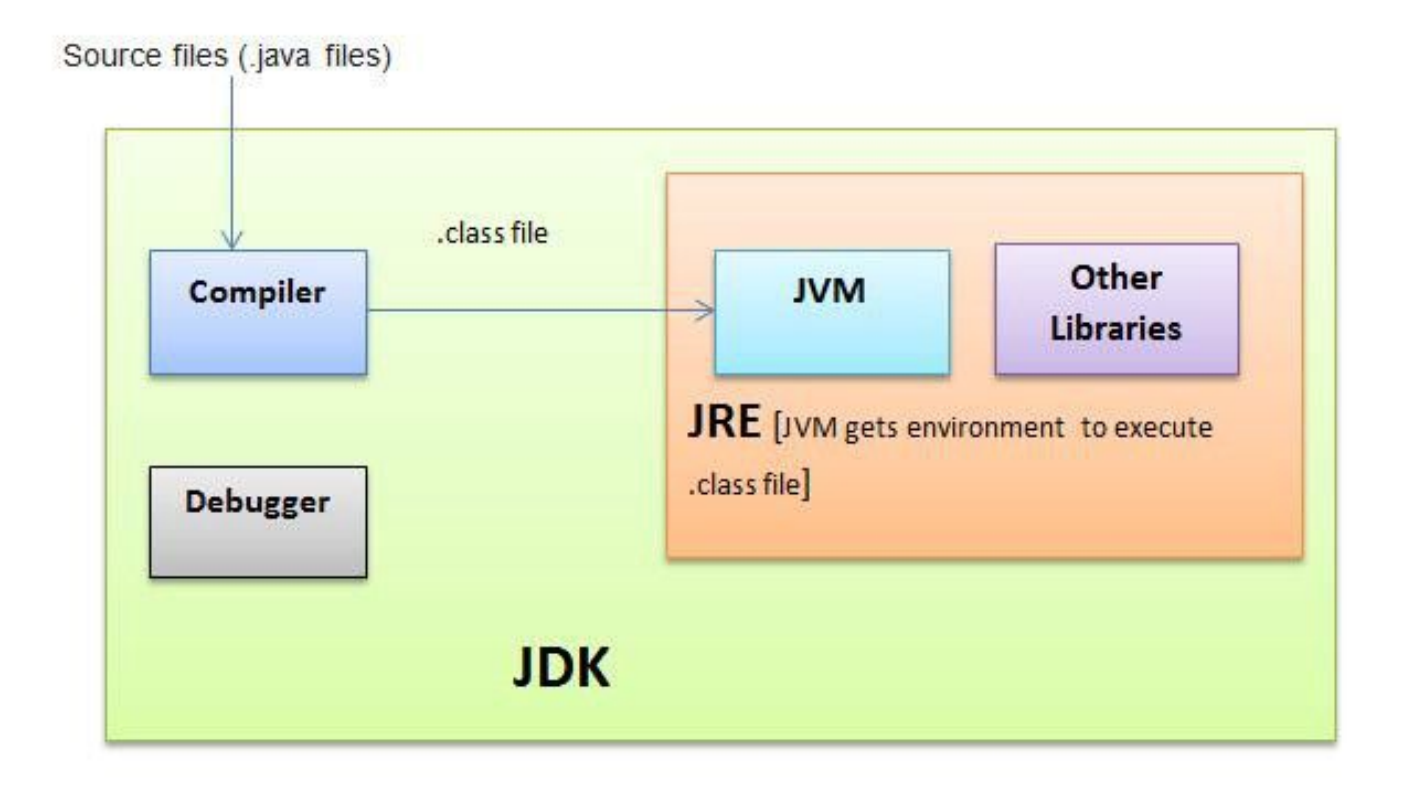
Software is like cathedrals. First we build them, then we pray.

Java is a programming language and computing platform.

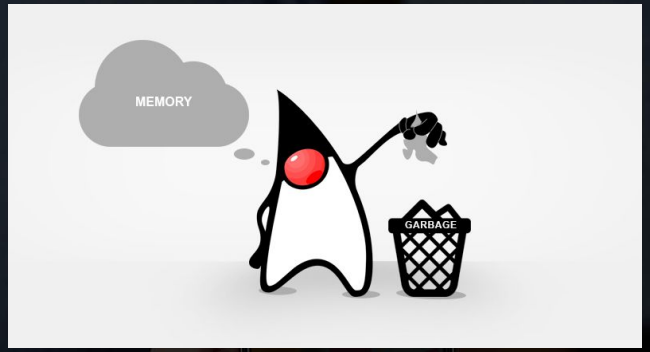
It includes:
- Java Programming Language
- Java Development Kit
- Java Runtime Edition
- Java Virtual Machine

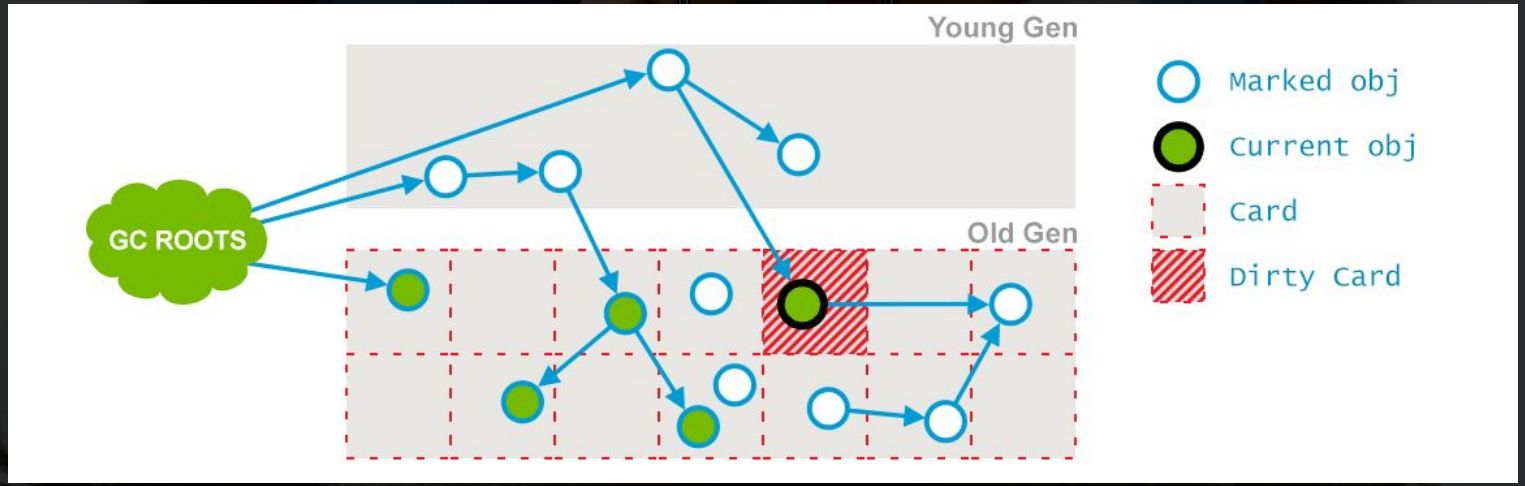


How Java Works?



How Java Works?





Concepts

Object:
- State
- Behavior
- Identity

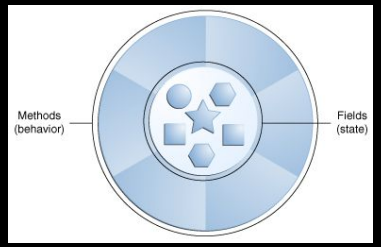
Class:
- Attributes
- Functions
- Instance

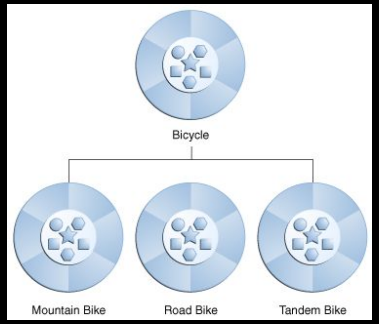
Inheritance:
- State and Behavior from Other Classes

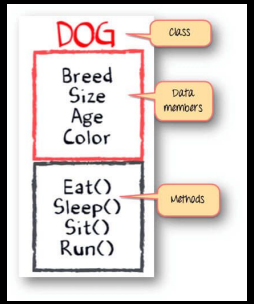
Interface:
- Achieve Abstraction
- Multiple Inheritance

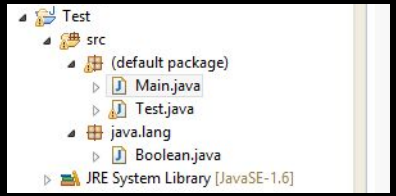
Package:
- Keep Organized and Categorized

Encapsulation:
- Self-contained objects

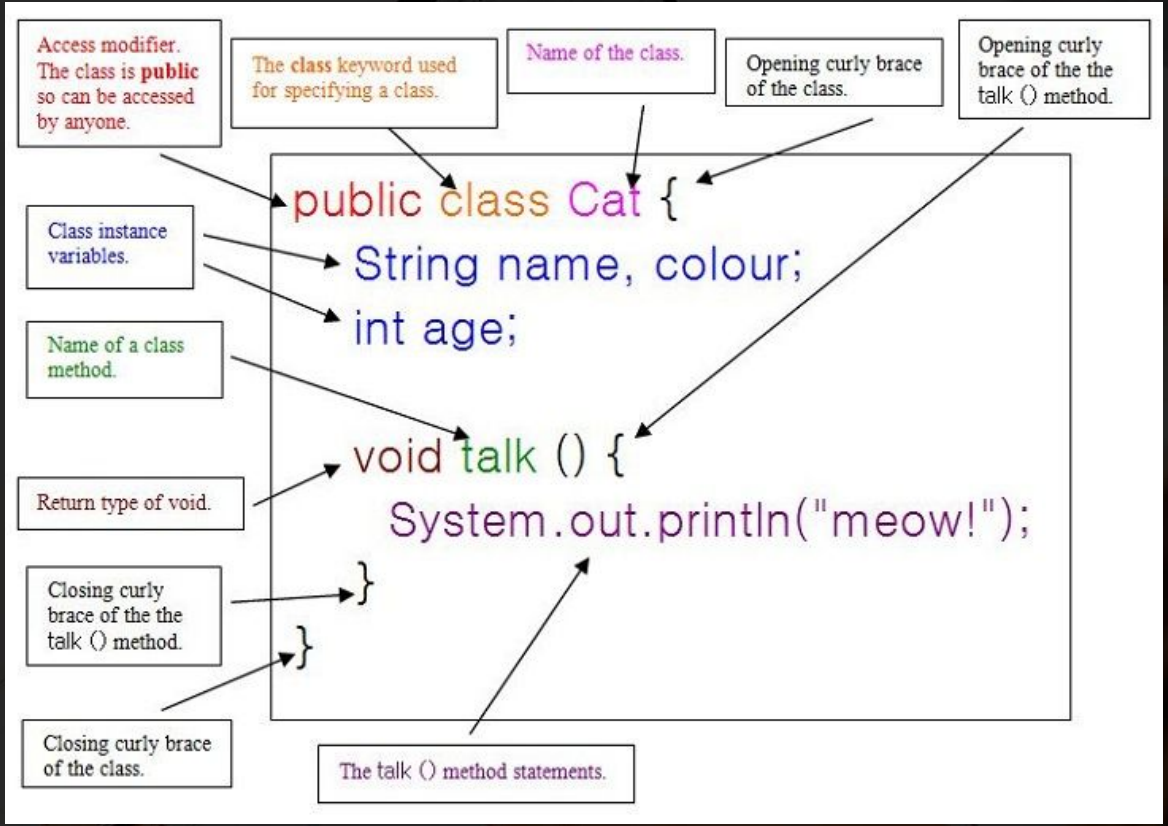




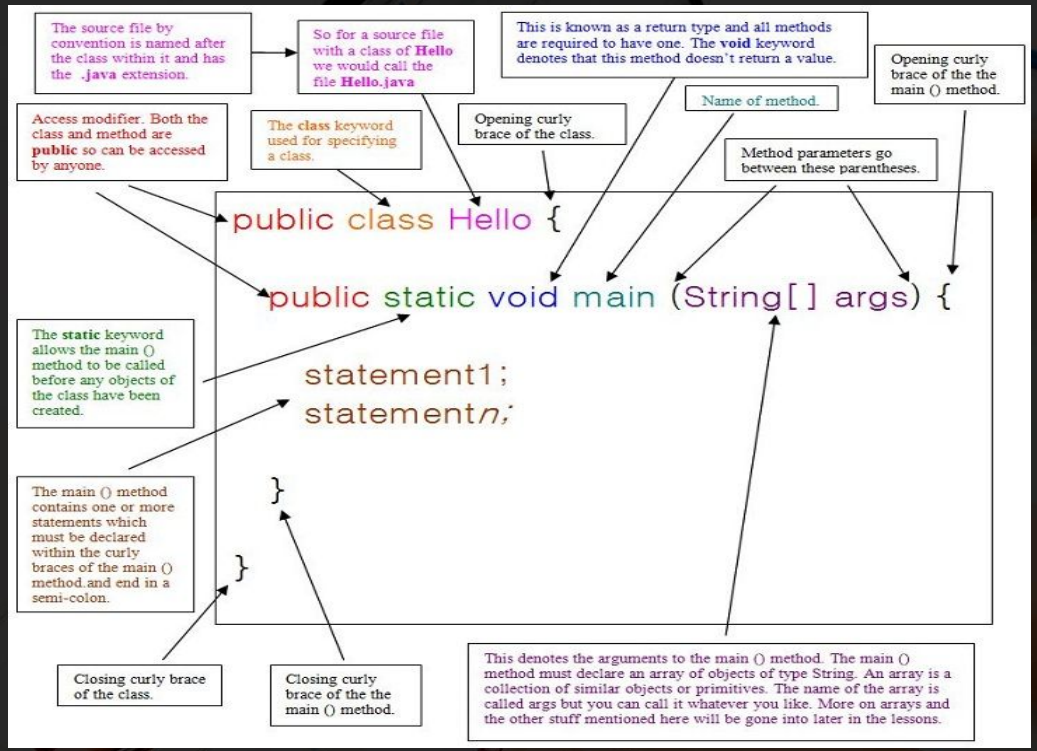




A Java Class


Access modifier. The class is **public** so can be accessed by anyone.
The **class** keyword used for specifying a class.
Name of the class.
Opening curly brace of the class.
Opening curly brace of the the talk () method.
public class Cat {
Class instance variables.
String name, colour;
int age;
Name of a class method.
void talk () {
Return type of void.
System.out.println("meow!");
Closing curly brace of the the talk () method.
}
}
Closing curly brace of the class.
The talk () method statements.

A Java (Main) Class



Java Language Concepts

- **Variables:**
  - Types
    - Primitive (int, char, byte, double, ...)
    - Non-Primitive (Object, String, Integer)
  - Static Variables
  - Constants (Instance) Variables
  - Arrays
- **Methods:**
  - Signature
  - Static Methods
  - Return Type (void, int, String...jaja)
- **Constructor:**
  - Instance
  - **super**
  - **this**
- **Accesor Modifiers:**
  - public
  - private
  - protected
  - Default -> *package*
- **Class:**
  - Types
    - public
    - final
    - abstract
    - Synchronizable
- **Packages:**
  - CLASS_PATH
  - Class Loader
  - Imports
- **Annotations:**
  - Is a kind Interface/Class

Design Patterns

In software engineering, a design pattern is a general repeatable solution to a commonly occurring problem in software design. A design pattern isn't a finished design that can be transformed directly into code. It is a description or template for how to solve a problem that can be used in many different situations.

- Creational
  - Class instantiation (create new objects)
- Structural
  - Class and Object composition
- Behavioral
  - Class's objects communication